UNIVERSITAT POLITÈCNICA DE CATALUNYA

Ph.D. Dissertation

Surface Reconstruction for Multi-View Video

Jordi Salvador Marcos

Thesis supervisor:

Dr. Josep R. Casas Pla

Department of Signal Theory and Communications
Universitat Politècnica de Catalunya

Barcelona, July 2011

ACTA DE QUALIFICACIÓ DE LA TESI DOCTORAL

Reunit el tribunal integrat pels sota signants per jutjar la tesi doctoral:

Títol de la tesi: .....
Autor de la tesi: .....

Acorda atorgar la qualificació de:

- No apte
- Aprovat
- Notable
- Excel·lent
- Excel·lent Cum Laude

Barcelona, ……. de/d'…..……. de .....….

El President

.....
(nom i cognoms)

El Secretari

.....
(nom i cognoms)

El vocal

.....
(nom i cognoms)

El vocal

.....
(nom i cognoms)

El vocal

.....
(nom i cognoms)

A la meva estimada Sibylle,
els meus pares, la Paquita i en Jordi,
la meva germana Noelia
i un amic que mai no oblidaré, en Jacky

Resum

A questa tesi presenta diferents tècniques per a la definició d'una metodologia per obtenir una representació alternativa de les seqüències de vídeo capturades per sistemes multi-càmera calibrats en entorns controlats, amb fons de l'escena conegut.

Com el títol de la tesi suggereix, aquesta representació consisteix en una descripció tridimensional de les superfícies dels objectes de primer pla. Aquesta aproximació per la representació de les dades multi-vista permet recuperar part de la informació tridimensional de l'escena original perduda en el procés de projecció que fa cada càmera.

L'elecció del tipus de representació i el disseny de les tècniques per la reconstrucció de l'escena responen a tres requeriments que apareixen en entorns controlats del tipus *smart room* o estudis de gravació, en què les seqüències capturades pel sistema multi-càmera són utilitzades tant per aplicacions d'anàlisi com per diferents mètodes de visualització interactius.

El primer requeriment és que el mètode de reconstrucció ha de ser *ràpid*, per tal de poder-ho utilitzar en aplicacions interactives. El segon és que la representació de les superfícies sigui *eficient*, de manera que en resulti una compressió de les dades multi-vista. El tercer requeriment és que aquesta representació sigui *efectiva*, és a dir, que pugui ser utilitzada en aplicacions d'anàlisi, així com per visualitació.

Un cop separats els continguts de primer pla i de fons de cada vista –possible en entorns controlats amb fons conegut–, l'estratègia que es segueix en el desenvolupament de la tesi és la de dividir el procés de reconstrucció en dues etapes. La primera consisteix en obtenir un *mostreig* de les superfícies (incloent orientació i textura). La segona etapa proporciona superfícies tancades, contínues, a partir del conjunt de mostres, mitjançant un procés d'*interpolació*.

El resultat de la primera etapa és un conjunt de punts orientats a l'espai 3D que representen localment la posició, orientació i textura de les superfícies visibles pel conjunt de càmeres. El procés de mostreig s'interpreta com un procés de cerca

de posicions 3D que resulten en correspondències de característiques de la imatge entre diferents vistes. Aquest procés de cerca pot ser conduït mitjançant diferents mecanismes, els quals es presenten a la primera part d'aquesta tesi.

La primera proposta és fer servir un mètode basat en les imatges que busca mostres de superfície al llarg de la semi-recta que comença al centre de projeccions de cada càmera i passa per un determinat punt de la imatge corresponent. Aquest mètode s'adapta correctament al cas de voler explotar foto-consistència en un escenari estàtic i presenta característiques favorables per la seva utilització en *GPUs* –desitjable–, però no està orientat a explotar les redundàncies temporals existents en seqüències multi-vista ni proporciona superfícies tancades.

El segon mètode efectua la cerca a partir d'una superfície inicial mostrejada que tanca l'espai on es troben els objectes a reconstruir. La cerca en direcció inversa a les normals –apuntant a l'interior– permet obtenir superfícies tancades amb un algorisme que explota la correlació temporal de l'escena per a l'evolució de reconstruccions 3D successives al llarg del temps. Un inconvenient d'aquest mètode és el conjunt d'operacions topològiques sobre la superfície inicial, que en general no són aplicables eficientment en *GPUs*.

La tercera estratègia de mostreig està orientada a la paral·lelització –*GPU*– i l'explotació de correlacions temporals i espacials en la cerca de mostres de superfície. Definint un espai inicial de cerca que inclou els objectes a reconstruir, es busquen aleatòriament unes quantes mostres llavor sobre la superfície dels objectes. A continuació, es continuen buscant noves mostres de superfície al voltant de cada llavor –procés d'expansió– fins que s'aconsegueix una densitat suficient. Per tal de millorar l'eficiència de la cerca inicial de llavors, es proposa reduir l'espai de cerca, explotant d'una banda correlacions temporals en seqüències multi-vista i de l'altra aplicant multi-resolució. A continuació es procedeix amb l'expansió, que explota la correlació espacial en la distribució de les mostres de superfície.

A la segona part de la tesi es presenta un algorisme de mallat que permet interpolar la superfície entre les mostres. A partir d'un triangle inicial, que connecta tres punts coherentment orientats, es procedeix a una expansió iterativa de la superfície sobre el conjunt complet de mostres. En relació amb l'estat de l'art, el mètode proposat presenta una reconstrucció molt precisa (no modifica la posició de les mostres) i resulta en una topologia correcta. A més, és prou ràpid com per ser utilitzable en aplicacions interactives, a diferència de la majoria de mètodes disponibles.

Els resultats finals, aplicant ambdues etapes –mostreig i interpolació–, demostren la validesa de la proposta. Les dades experimentals mostren com la metodologia presentada permet obtenir una representació ràpida, eficient –compressió– i efectiva –completa– dels elements de primer pla de l'escena.

Summary

This thesis presents different techniques for the definition of a methodology for obtaining an alternative representation of video sequences captured by calibrated multi-camera systems in controlled environments, with known scene background.

As suggested by the title of the thesis, this representation consists in a three-dimensional description of the surfaces of foreground objects. This approach for the representation of multi-view data allows for the recovering of part of the three-dimensional information of the original scene lost int the projection process happening in each camera.

The choice of a type of representation and the design of the techniques for 3D scene reconstruction are driven by three requirements that appear in controlled environments such as a *smart room* or recording studios. In these scenarios, video sequences captured by a multi-camera rig are used both for analysis applications and interactive visualization methods.

The first requirement is that the reconstruction method must be *fast* in order to be usable in interactive applications. The second one is that the surface representation must be *efficient*, providing a compression of the multi-view data. The third requirement is that this representation must be *effective* or, in other words, that it provides the relevant information to be used for analysis applications as well as for visualization.

Once separated foreground and background elements from each view –possible in controlled environments with known background–, the strategy that is followed in the development of the thesis is that of dividing the reconstruction process in two stages. The first one consists in obtaining a *sampling* of the foreground surfaces (including orientation and texture). The second stage provides closed, continuous surfaces from the set of samples, through an interpolation process.

The result of the first stage is a set of oriented points in 3D space that locally represent the position, orientation and texture of surfaces that are visible from

multi-camera settings. The sampling process is interpreted as a search for 3D positions that result in feature matchings between different views. This search process can be driven by different mechanisms, which are presented in the first part of this thesis.

The first proposal is to use an image-based method that searches surface samples along the half-line which starts at each camera's center of projections and goes through a certain point in the corresponding image. This method is well-suited to exploiting photo-consistency in a static scenario and presents favorable features for its usage in *GPUs* –desirable–, but it is not oriented towards the exploitation of temporal redundancies in multi-view sequences, nor provides closed surfaces.

The second method performs the search starting from an initial sampled surface which encloses the space where the objects to be reconstructed lie. The search, in the inverse direction of the normals –pointing towards its interior–, allows to obtain closed surfaces with a algorithm which exploits the correlation in the temporal sequence of reconstructions of the scene. A drawback of this method is the set of topological operations over the initial surface, which are in general not efficiently applicable in *GPUs*.

The third sampling strategy is oriented towards parallelization –*GPU*– and the exploitation of spatial and temporal correlations in the search of surface samples. Defining an initial search space that includes the objects to be reconstructed, some seed samples are randomly scouted, which lie on the surfaces of the objects. Next, new surface samples are searched for around each seed –expansion process– until a sufficient density is achieved. In order to improve the search efficiency for seed samples, a reduction of the search space is introduced, either by exploiting temporal correlations in multi-view sequences or by applying multi-resolution. Next, the algorithm proceeds with the expansion, which exploits the spatial correlation in the distribution of surface samples.

In the second part of the thesis, a meshing algorithm is presented, which allows for the interpolation of the surfaces between its samples. Starting by an initial triangle, which connects the points coherently oriented, an iterative expansion of the surface over the complete set of samples takes place. Compared to the *state-of-the-art*, the proposed method presents a very accurate reconstruction (it does not modify the position of samples) and results in a correct topology. Furthermore, it is fast enough as to be usable in interactive applications, as a difference with the majority of available methods.

The final results, applying both stages –sampling and interpolation–, reflect the validity of the proposal. Experimental data show how the presented methodology permits obtaining a fast, efficient –compression– and effective –complete– representation of the foreground elements of a scene.

Acknowledgements

I would like to gratefully acknowledge all the people who have contributed to the completion of this dissertation, either by discussing the technical concepts related to it or by giving me their unconditional support at all times. Without them, the result of this thesis at its completion would have not been nearly as satisfying.

In first place, I can only express my highest gratitude for the initial guidance and final review provided by each of the reporters of this thesis: Jean-Sébastien Franco (INP Grenoble University and INRIA Rhône-Alpes), Francisco Morán (Universidad Politécnica de Madrid), Oliver Schreer (Fraunhofer Heinrich Hertz Institute Berlin) and Aljoscha Smolic (Disney Research Zürich).

I also want to thank all the people in the Image and Video Processing Group of the Universitat Politècnica de Catalunya, with whom I had the pleasure of working during the last years. I would like to emphasize my gratitude to those colleagues and professors who influenced my work during the development of this thesis, including Marcel Alcoverro, Cristian Canton, Jaime Gallego, Albert Gil, Adolfo López, Montse Pardàs, Jordi Pont and Xavi Suau. I want to add to this list two exceptional engineers with whom I also had the opportunity of working by supervising their master's theses, Enrique Oriol and Marc Maceira, and all the people who made me remember how nicer life can be when you make a small break to play some football, starting with David and Pedro and continuing with the rest of players.

My family and friends have also been fundamental for making the development of this thesis worthwhile. I thank my parents Paquita and Jordi and my sister Noelia, my grandmother Paquita, Axel and Fabiola and all my other relatives for their unconditional support. I also want to thank my second family, Dieter and Chrsitine, Philip, Matthias, Ulrike, Lutz and Tilman and of course all of my friends.

I want to specially thank my girlfriend Sibylle for all the support she gave me in both the good and the not-so-good times. Without her, many things in my life would never be as good as they are.

Last but not least, I want to attribute the success of the completion of this thesis to Josep R. Casas. Apart from having introduced me to the world of research and having guided me with his conviction and positive way of thinking in many difficult moments, he has also been a friend. For all these things I cannot but show my highest gratitude and admiration.

Mannheim
July 2011

Contents

List of Figures

Chapter 1

Introduction

Over the last years, analysis applications exploiting visual information have evolved from environments where scenes are captured by a single camera to setups where several cameras provide multi-view image sets. Newer imaging technologies provide also more advanced features, such as depth, but they still lack the interference-free, placing flexibility of standard video cameras.

Thus, the main advantage of multi-view scenarios is the availability of richer information about the actual 3D scene, when compared to imaging a scene from a single viewpoint. What is lost in the imaging process is, precisely, the 3D structure of the scene. This is due to the projection onto a 2D image on the camera sensor offering the perspective of a single view-point. Multi-view systems are able to recover 3D information from 2D projections in multiple viewpoints. This information allows for a richer representation of the scene from which both visualization and analysis applications can benefit. For example, objects affected by occlusions when being observed from a certain viewpoint can be observed without ambiguity by using data from other viewpoints. Furthermore, redundancies among different views contribute positively by enhancing the robustness of the scene representation, either by resolving inconsistencies in detected features or by adding 3D location information.

A practical disadvantage in a multi-view scenario is the management cost of the richer available data. An obvious problem in multi-view scenarios is that the storage –or transmission, in a streaming application– of a large number of video sequences that contain the projections of a certain scene from each of the available viewpoints is not negligible. Indeed, if a very large number of views is considered, the situation may become intractable. An important observation is that this cost should be considered unjustified, because the multiple sequences captured by each camera present redundancies –the larger number of available views, the more redundant the multi-view data is–.

The target of this thesis is to present an efficient method for representing this multi-view information, at the expense of a certain, reasonable computational cost, providing an alternative representation for visualization, analysis, transmission and storage. The computational resources required by such an efficient representation should not increase linearly with the number of views, regarding the information redundancy of multi-view data. Another requisite of such a representation is that it allows to go back to the original views, given the fact that it conveys all the information contained in the latter.

This representation is the result of the 3D reconstruction methodology that is proposed in this thesis. The main idea is not only to reduce the storage and transmission costs related to the multi-view data, but also to do that in an efficient way, such that it is adequate to be used by both visualization and analysis applications. Regarding this last point, the 3D representation of the multi-view data should enrich the possibilities of such applications. Some features –*e.g.* surface normals– are not easily computable from the multiple views, whereas their computation can be a relatively simple task given a suitable 3D representation. To sum up, using a 3D representation we obtain efficiency (compression) and effectiveness (adaptation to visualization and analysis) in order to represent the rich information available in multi-view setups.

1.1 Context

The target scenarios of this thesis are controlled environments. Examples of these are *smart rooms* –or *smart spaces*– and recording studios. In the former, a varying number of people or objects can interact with each other as well as with computer systems. This sets the adequate scenario for the development of services related to advanced human-computer interfaces (HCI) not requiring special attention by the user, as a the typical HCI display-keyboard-mouse would require. In the latter, recording studios, multi-view techniques can be exploited also for video production.

In these environments, video cameras –as well as microphone arrays and other capture equipment– are distributed around the scene in an almost non-invasive manner, usually in rather wide-baseline configurations. In many cases, the contents of the background of the scene can be assumed to be static, or are even removed –*chroma keying*–. We consider this case, and focus on the foreground –*i.e.* the dynamic part– of the scene.

The projection process that takes place in each camera in order to generate a 2D image out of the actual 3D scene is an inherently lossy process, for most of the 3D information about the scene is lost. An image-based representation of the multi-view data provides cues that can be used to recover part of this 3D

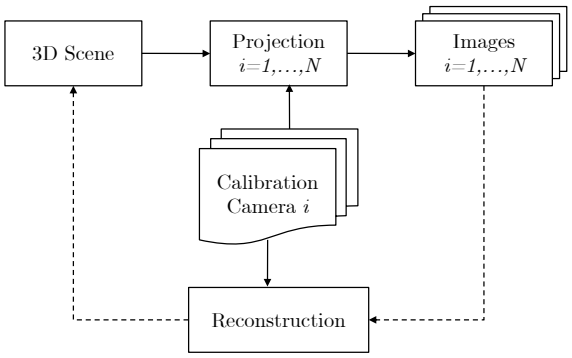


Figure 1.1: 3D reconstruction as a recovering mechanism for the 3D cues lost in the imaging process. The result of the reconstruction process is an approximation to the actual geometry and texture of the 3D scene, from which only some surfaces are visible

information. However, it is the process of multi-view reconstruction the one that explicitly attempts to recover the information lost in the projection process in order to provide a richer representation, at least of some parts of the scene. This concept is illustrated in Fig. 1.1.

Hence, the adopted approach is that of obtaining a unique, 3D representation of the foreground contents of the scene captured by such a multi-camera rig. In more detail, the main subject that this thesis addresses is the *multi-view reconstruction* of the foreground elements of scenes in the aforementioned scenarios with known background. The purpose of such 3D representation is two-fold, since it is to be used by different algorithms exploiting the enriched information obtained by multi-camera rigs for enriched visualization –*free-viewpoint video* [Lipski et al., 2010]– and also for analysis of the contents of the scene.

1.1.1 Challenges

Solutions to several problems in multi-view reconstruction have already been proposed, which provide different levels of accuracy depending on the amount and nature of the available information. Thus, whereas there exist techniques able to compete with range scanners in terms of accuracy [Seitz et al., 2006], they require a small enough baseline in order to determine the 3D structure of the scene being reconstructed. On the other extreme, there also exist techniques which, although being able to obtain 3D representations from a small number of viewpoints in a much wider baseline configuration, they are limited by the accuracy or the robustness with respect to inconsistencies in the input data they can provide.

In this thesis, we discuss the suitability of a surface-based representation of the 3D data as opposed to a volumetric one in terms of its efficiency at describing the visible features in multi-view scenes. Continuous surface representations have been

widely used in computer graphics, resulting in powerful graphics hardware able to handle triangle mesh representations of 3D surfaces. Also sample-based descriptions of surfaces can be considered when the sampling density is high enough and an explicit description of the topology is not required.

Few methods are available –EPVH in [Franco and Boyer, 2003] being perhaps the clearest example– which are capable of obtaining a triangle mesh directly from image-based features, without inputting explicit 3D estimates to the location of some surface points. However, many techniques exist –*e.g. Poisson reconstruction* [Kazhdan et al., 2006]–, which are capable of robustly obtaining triangle meshes from sets of oriented surface points. One drawback of the approach consisting in computing the 3D position of surface points and obtaining a triangle mesh by using such a technique is that the computation time for the mesh might be too large for being used on-line.

Whereas the former approach has already been used for real-time operation on adequate hardware configurations, the latter seems more suitable in order to provide a surface description useful for both analysis and visualization. This is due to the more uniform distribution of the size of the triangle faces composing the resulting surface description and the higher density of vertices, which might introduce an additional rendering cost but provides richer information about local features of the surface –*e.g.* orientation or curvature–.

It is also interesting to note that most works in 3D reconstruction target static scenes, *i.e.*, scenes where no more than one single frame is captured –for multi-view settings using multi-camera rigs– or scenes where a static object is captured at different time instants from different viewpoints. Few authors, [Vedula et al., 1999, Starck and Hilton, 2005] among others, have addressed the problem of exploiting the redundancies between consecutive time instants in video sequences to either improve the quality or reduce the computational cost of 3D reconstruction.

There are three main challenges for the 3D reconstruction methodology proposed in this thesis. The first one is that its result must be able to be used as a replacement of the multi-view data without introducing a significant loss of information. The second one is that it has to be efficient in terms of its representation, resulting in a compression of the multi-view data. Finally, it has to be effective in order to facilitate the posterior use of the data in visualization and analysis applications.

Going slightly more in detail, some challenges regarding the design of new algorithms for surface-based 3D reconstruction can also be detected. These algorithms must provide a sufficient level of accuracy –projection of the 3D shape *vs.* original image– without restrictions to the baseline. In second place, it is necessary to obtain a throughput such that the reconstructed surface can be used for real-time visualization and analysis applications and, in third place, it is also necessary, in

terms of efficiency and effectiveness of the approach, to adapt the 3D reconstruction to the case of multi-view video sequences in order to exploit temporal correlations.

1.2 Objectives

Regarding the previously described challenges in the context of 3D reconstruction of the surfaces of foreground objects in multi-view video sequences, the main objective of this thesis is to provide a unique surface-based representation of the relevant multi-view data under the aforementioned conditions:

- The resulting surface representation must be efficient –compressing the redundancies in multi-view information–, effective –suitable for visualization and analysis– and accurate –minimizing losses with respect to the original data–.
- It must be able to retrieve a close approximation to the original images –with minimal losses– by re-projection of the unique surface-based representation and addition of the background information.
- It should exploit both spatial and temporal correlation in multi-view video sequences in order to reduce the computation time and make it suitable for real-time operation.
- Finally, another goal is to obtain an efficient, continuous representation of the surface –a triangular mesh– in a usable computation time. Such a representation can contribute to further compress the multi-view data and can be used for analysis as well as for rendering using standard graphics pipelines.

As a summary, the ultimate goal of this thesis is *to obtain a fast –towards real-time–, efficient –within a compressed/limited support–, effective –exploitable by forthcoming applications– and unique surface-based representation of each frame in multi-view video sequences, which provides all the relevant information about foreground objects contained in the original streams and can be used for both interactive visualization and analysis applications.*

1.3 Structure

Following this brief introduction to the challenges that are tackled in this thesis and the goals that are to be achieved, the reader will find a review of the related state of the art in Chapter 2. Next, Chapter 3 presents the approach that is followed throughout this thesis, which is divided in two main concepts.

The first main concept is the dense sampling of surfaces from multi-view data –3D reconstruction–. This is presented in Part I as an evolutionary collection of different techniques, each with its advantages and its drawbacks.

The second main concept is the interpolation of surfaces from a dense set of samples –meshing–. The proposed algorithm for this purpose and a comparison to other methods in the literature are presented in Part II.

In Part III, a validation of the objectives of the thesis is presented. These results are followed by the final discussion about the achievement of the proposed goals and the resulting contributions.

Chapter 2

State of the Art

Applications derived from research in the exploitation of the rich data available in multi-view scenarios range from 3D reconstruction of monuments and buildings [Mueller et al., 2004] to volumetric computer tomography [Sardarescu et al., 2008]. Computer vision algorithms benefit of the enriched information available in such scenarios, as well. In general, visual analysis from multiple views exploits the similarity and disparity features present in the available projections of the scene.

Similarity features (redundancies) that can be identified from different viewpoints allow for the establishment of correspondences for the registration of the images in a common geometric framework. This allows for an easier detection and analysis of these redundancies in the several views. On the other hand, the disparity features that can be detected in the multiple views contribute to augment the visual information available through the inclusion of visual elements belonging to areas that are hidden in one of the views, thus disambiguating occlusions.

One of the possibilities when working in multi-view scenarios consists in reconstructing the 3D shapes of elements in a scene that has been captured by a multi-camera setting. This approach appears in opposition to working in an image-based manner, using geometric relationships between pairs of images described by epipolar constraints. Therefore, multi-view approaches can be classified in:

- **3D reconstruction**. Three-dimensional shapes are computed from the multi-view data, with several possibilities for their representation, each providing different advantages and drawbacks, as seen below.
- **Image-based multi-view**. Analysis is performed directly on the captured images and multi-view cues are exploited by considering epipolar constraints that can be computed from *fundamental matrices* (Appendix A).

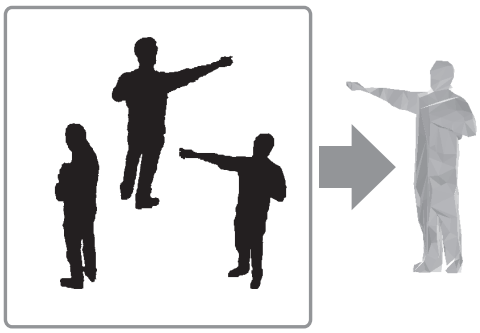


Figure 2.1: Visual Hull of a set of silhouettes. Only foreground/background classification information is used to reconstruct the 3D shape

Thus, a detailed description of the visual data in multi-view settings, for applications ranging from visualization to analysis –*e.g.* object manipulation or visual recognition–, might be based on the reconstruction of 3D models of the environment and of the objects present in it. Such reconstruction will be more precise as more cameras observe the space where the scene is located.

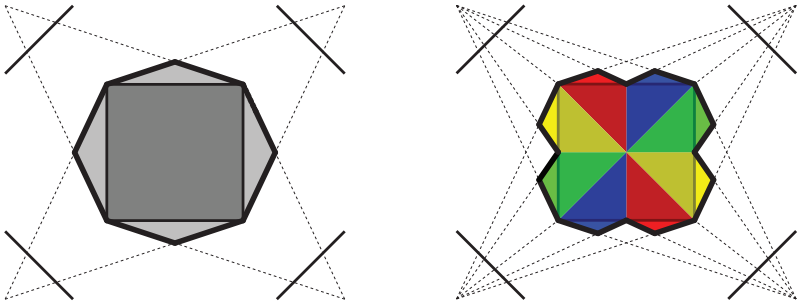
We focus this dissertation on 3D reconstruction to obtain a compact and meaningful representation of multi-view data. This can be done efficiently and contribute to the state of the art. Besides, this representation can be used to freely choose the viewpoint for visualization applications. Real-time analysis applications can also benefit of the enriched information available in settings with an increasing number of viewpoints, without necessarily increasing the computational load as would result in image-based scenarios, involving comparisons or consistency checks. Instead, a unique, compact representation of the visual cues provides a generic framework that isolates visualization applications or high-level analysis from the low-level details of the multi-view capture process.

2.1 Multi-View 3D Reconstruction

In contrast with image-based multi-view approaches, 3D reconstruction methods provide a representation with a 3D support that can be used to explain in a single, compact structure the multi-view cues from the input images with an arbitrary level of detail, limited by the baseline configuration of the multi-view setup.

2.1.1 3D reconstruction taxonomy

There are several techniques for computing 3D reconstructions from a set of images captured in multi-camera settings. These techniques can be classified accordingly to several differentiating properties. In the following, two possible classifications,



(a) (b)

Figure 2.2: Tightness of the photo hull compared to the visual hull. In (a), the bold line represents the visual hull of a 2D scene (in this case, the central occupied square) reconstructed from a set of silhouettes (segments) available in a set of 1D cameras. In (b), the photo hull is computed and represented by the bold line, resulting in a tighter reconstruction of the actual shape, which is the colored square

each regarding a certain differentiating property, are listed.

Classification by type of input data. Depending on the type of input data, we can find different types of 3D reconstruction methods in the literature:

- **Visual hull**. In [Laurentini, 1994], the *visual hull* was introduced as the maximal 3D volume which is consistent with the silhouettes of an object projected onto a set of images. In this case, input data consist solely of binary silhouettes. This type of 3D reconstruction is depicted in Fig. 2.1.

- **Photo hull**. In [Kutulakos and Seitz, 1999], the *photo hull* was introduced as the maximal 3D volume which is photo-consistent with each image. In this case, input data are the captured color images for a given time instant. Due to the richer information available in the input data, the photo hull constitutes a tighter estimate of the actual volume than the visual hull, being its lower robustness, in comparison with that of the visual hull, a drawback to consider when attempting to use it for analysis applications. The tightness of the photo hull, compared to that of the visual hull, is illustrated in Fig. 2.2.

- **Multi-view stereo**. Similarly to the photo hull computation, in multi-view stereo [Goesele et al., 2006] the input images captured by the cameras are considered for reconstruction. However, in this case a small baseline camera setup is required in order to precisely compute the position of surface points that match a photo-consistency test between pairs of images. Typically, normalized cross-correlation is used as a photo-consistency test, either in grayscale or in the RGB color space.

Classification by representation support. The majority of 3D reconstruction algorithms in the related literature rely on one of the following representations:

- **Volumetric.** In many algorithms, for example in [Szeliski, 1993], a discrete occupancy function over a regularly sampled 3D grid –**voxelization**– is used to describe the 3D structure of the reconstructed scene. Whereas this support is useful to represent three-dimensional dense volumes, it appears as a poor descriptor of the visible data in multi-camera settings, which are the projections of object surfaces. In the cited reference, a progressive algorithm allows to obtain the occupancy function in homogeneous areas without having to visit every voxel of the highest-resolution regular grid individually (these are exclusively visited when needed).

  In [Faugeras and Keriven, 2002], **level sets** were introduced. They are functions encoding distances of elements in a regularly sampled 3D grid to their closest surfaces. The advantage of methods based on the level set representation is that numerical computations involving curves and surfaces can be computed without having to parameterize these objects. Furthermore, methods based on level sets provide an effective manner of following shapes that change their topology, making level sets a convenient representation for modeling time-varying objects. An illustration of such representation is shown in Fig. 2.3.

- **Surface-based.** Due to their efficiency for tessellation of surfaces, **polygon meshes** are a popular output format for multi-view algorithms. They have also been used as the central representation by some authors, for example in [Isidro and Sclaroff, 2003, Franco and Boyer, 2003]. As a difference with the previous representation supports, polygon meshes describe only the geometry of surfaces, and not their interior. Thus, it is a representation better suited to the data available in multi-camera settings. An advantage with respect to the previous representations is that 3D space does not need to be discretized in order to obtain it.

  Surface patches, also called surfels or oriented points in some references in the literature [Jia et al., 2006, Carceroni and Kutulakos, 2002], are sampling elements designed to describe 3D surfaces. As such, they basically consist in a 3D location and a 3D orientation descriptor. They constitute a suitable support for describing surfaces of the objects in a multi-view scene, since they efficiently describe the only features visible by the cameras.

- **Image-based.** A **depth map** representation consists in setting a depth value (distance to image) for each pixel in each available view. It has been used, for example, in [Kolmogorov and Zabih, 2002, Kauff et al., 2007]. This representation, compared to the others, lacks a 3D support, which makes its

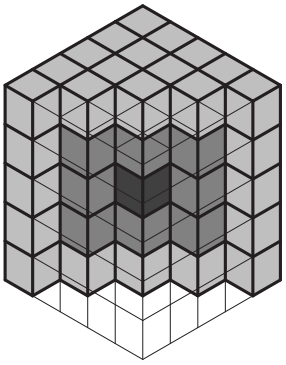


Figure 2.3: Level sets of a voxelized solid cube. In this example, where a cut of a solid cube is shown, gray levels would encode the minimal distance of each voxel to the surface of the reconstructed cube

usage in analysis applications potentially more costly, due to the increasing computational demands of multi-view matching with elevate numbers of available input views. However, depth maps can be an efficient representation for classical stereo or tri-focal with narrow-baseline [Waizenegger et al., 2011].

Additionally, in [Casas and Salvador, 2006], a discrete occupancy function over an irregular 3D grid, sampled in a specific way such that it appears as adapted to the calibration geometry of the input images, is also shown as an intermediate step between volumetric and image-based representations because, like the latter, the image-adapted sampling scheme does not introduce an additional, arbitrary sampling of 3D space that would result in sampling artifacts like those appearing in volumetric techniques, whereas, like the former, it keeps a sampled representation that can potentially expand current algorithms for processing voxelized volumes.

2.1.2 Visual hull

As introduced above, the *visual hull* is the maximal volume consistent with a set of input silhouettes. Depending on the formation model of the input silhouettes, different techniques allow the estimation of the visual hull, each of them using one of the representation supports introduced above.

Shape from silhouette. *Shape from silhouette* (SfS) techniques require the previous computation of binary silhouettes of foreground objects in each of the available views. In this case, the computed silhouettes are considered noiseless, thus not needing for special care beyond that of geometrical constraints when checking consistencies between views. Reviewing the literature, we report the following approaches:

- **Voxelized Shape from Silhouette.** A multi-resolution technique based in *octrees* [Meagher, 1982, Szeliski, 1993] is a widely extended approach. In it, a coarse representation of the visual hull is obtained and, at subsequent iterations, the resolution of the representation is increased in parts of the volume that require higher sampling frequencies for obtaining a suitable representation without excessive aliasing (surfaces). Another technique, with fixed resolution for all voxels but computationally efficient in the definition of its projection test, consists in randomly sampling the actual projection of a voxel on the images, rather than using the whole *splat* onto which a voxel projects for a given point of view. This projection test is called SPOT (*Sparse Pixel Occupancy Test*), and was introduced in [Cheung et al., 2000].

- **Implicit Surface Visual Hull.** This technique attempts to achieve a higher reconstruction quality by the adoption of a specific adaptive sampling scheme [Erol et al., 2005] that, similarly to the octree-based voxelization, uses a finer resolution in those regions where it is needed. As a second step, it is accompanied by a post-processing aiming at obtaining crack-free polygonal surfaces, using *marching cubes* [Lorensen and Cline, 1987].

- **Polyhedral Visual Hull.** This technique directly computes the 3D surface of the visual hull and describes it as a polygon mesh. Different views of this approach can be found in [Matusik et al., 2001] and [Lazebnik et al., 2007], [Franco and Boyer, 2003] or [Franco and Boyer, 2009]. Furthermore, [Franco et al., 2006] introduces the *visual shape* concept. Visual shapes are a class of silhouette-consistent shapes including, but not limited to, the visual hull. They are useful when observing shapes with known properties, such as smoothness, because this prior knowledge of the geometrical structure can be used for retrieving shapes with an appearance closer to the actual ones.

- **Compressed Sensing Reconstruction**. The application of the *compressed sensing* theory [Donoho, 2006] to the field of 3D reconstruction allows obtaining the volumetric occupancy from a smaller number of random projections, thanks to the exploitation of the sparse structure of the resulting occupancy function. This approach allows for the fast reconstruction of the visual hull, rendering the computation time practically independent of the number of input views, as shown in [Reddy et al., 2008]. The reconstruction of the actual scene occupancy from the sparse set of samples requires the efficient computation of the so-called *Basis Pursuit* [Chen and Donoho, 1994], a widely studied linear programming problem, which is considered tractable under the data structure assumptions in the compressed sensing theory.

Shape from silhouette with enhanced robustness. In a more general case, silhouette formation must be considered noisy, due to limitations in silhouette ex-

traction algorithms. Hence, consistency tests between views and further processing must address this problem. There are several approaches in the literature that can be grouped as follows:

- **Minimization of energy functions.** An energy function is defined, including functionals based on the local neighborhood structures of three-dimensional elements and smoothing factors, like in [Kolmogorov and Zabih, 2004] and [Alcoverro and Pardàs, 2009]. Algorithms based on *graph cuts* allow to obtain a global minimum of the defined energy function ([Kolmogorov and Zabih, 2002]) with great computational efficiency. The optimal graph cuts are obtained by first constructing a directed graph with a reversed edge for each edge between two nodes. The maximum flow algorithm is executed on the resulting graph, with each node representing a binary variable, and the resulting labeling of the nodes represents the minimum cut of the graph derived from the defined energy function.

- **Space occupancy grids** and other Bayesian frameworks. With *space occupancy grids* [Franco and Boyer, 2005], each pixel is considered as an occupancy sensor, and the visual hull computation is formulated as a problem of fusion of sensors with Bayesian networks. This method allows to avoid obtaining error-prone binary silhouettes by, instead, computing an occupancy probability for each pixel that is fused with the probabilities of other pixels in other views in order to set the voxel occupancy probability. This voxel occupancy probability can be thresholded at a final step in order to set the binary occupancy values.

- **Shape from Inconsistent Silhouette.** For cases when silhouettes contain systematic errors, [Landabaso et al., 2006] proposes a framework to minimize the classification error of a voxel when inconsistent views are a minority. To achieve that, the *inconsistent hull* is obtained, representing those voxels that are potentially occupied although they do not belong to the visual hull due to inconsistencies between silhouettes. The *unbiased hull* computed from the inconsistent hull, which represents voxels that should be part of the reconstructed volume, is added to the visual hull, resulting in a reconstruction robust to systematic errors in the silhouette extraction process.

Additionally, in [Shin and Tjahjadi, 2008], the *local convex hull* is introduced. This technique addresses some problems related to discontinuities, derived from the surface triangulation via *marching cubes* of octree-based voxelized representations of visual hulls. It proposes slicing an initial reconstructed volume for, in subsequent steps, generating local convex surfaces that are combined to obtain a more robust estimate of the surfaces of foreground objects.

2.1.3 Photo hull

As introduced above, when, instead of binary silhouettes, color images captured by multi-camera settings are used for reconstructing a scene, the *photo hull* can be obtained, representing the largest photo-consistent volume. In the literature we can find two different approaches for obtaining the photo hull of a scene in arbitrarily wide-baseline setups:

- **Voxel coloring.** In the algorithm proposed in [Seitz and Dyer, 1997], starting with an empty volume, voxels are visited in a specific order that provides consistent visibility for all the cameras and added to the reconstructed volume if they pass a photo-consistency test. One of its drawbacks is the need for a certain order for visiting the voxels, in order to ensure visibility. Furthermore, it assumes surfaces to be Lambertian, *i.e.*, they only reflect the diffuse component of the ambient light, making the appearance of surfaces independent of the point of view. Unfortunately, this assumption, made by many algorithms exploiting photo-consistency, is in general not true in real cases, which makes such methods much less robust than silhouette-based ones.

- **Space carving.** As a difference with the previous one, this algorithm [Kutulakos and Seitz, 1999] consists in setting as occupied a whole initial volume, which includes the actual scene, and iteratively removing those voxels that, visited in a specific order (typically using *sweep planes*), do not pass a photo-consistency test. Again, the Lambertian surface assumption appears in this approach, which keeps this method from achieving the robustness of silhouette-based ones.

2.1.4 Multi-view stereo

When small-baseline setups are available, finer photo-consistency tests can be used that take into account data in a vicinity of each pixel, assuming that perspectives will be similar between views. This set of techniques can be considered as an extension of classical stereo vision to a multi-camera environment, and introduces the problem of the appearance of inconsistent estimates between different pairs of views.

Thus, different multi-view stereo techniques introducing different types of refinements and solutions to the consistency problem can be found in the literature:

- **Depth-map extraction and fusion.** In [Goesele et al., 2006], depth-maps are extracted from each view by finding correspondences with, at least, two neighbor views, and computing its correspondent 3D position by triangula-

tion. Then, a volumetric combination of the depth-maps and posterior surface extraction is applied in order to obtain the photo-consistent reconstructed surface.

- **Energy minimization.** Volumetric min-cut methods (that are obtained from the maximum flow algorithm for graphs with reversed edges) combine photo-consistency with shape smoothing factors, as in [Sinha and Pollefeys, 2005]. More recently, [Campbell et al., 2008] considers several hypothesis for the placement of the actual surface, and an optimization step using Markov Random Fields gives impressive results for highly textured scenes. Its main drawback is that the method is unable to infer the surface position for objects that are not highly textured. This algorithm can be viewed as an improvement over [Goesele et al., 2006] using the same working principle.

- **Fusion with silhouettes**. In this case, the robustness of shape from silhouette approaches is combined with the availability of richer features in multi-view stereo by minimizing a convex functional with the exact silhouette consistency imposed as a convex constraint that restricts the domain of admissible surfaces to those that fit in the binary silhouettes once projected onto the original viewpoints, as in [Kolev and Cremers, 2008]. The authors claim high-accurate, silhouette-consistent reconstructions for real-world data in a computationally efficient manner.

- **Surface reconstruction from dense stereopsis**. In [Furukawa and Ponce, 2007], an initial sparse set of surface elements (surfels, surface patches or oriented points) are obtained from matching image-based features of several input images. From this initial estimation, an iterative algorithm augments the number of surface elements and rejects outliers, achieving results with high accuracy, even for small image sets. At a post-processing step, a closed polygonal mesh is morphed until it fits the set of reconstructed surface elements with an energy minimization algorithm. This method has the drawback of a high computational cost, with its bottleneck in the iterative stage that increases the density of surface elements.

2.2 Multi-View Video Reconstruction

One of the fundamental properties of a scene is its motion. Exploiting the temporal correlations of a scene is a task in computer vision that can benefit of the enriched information available in multi-view scenarios. Depending on the methodology applied for exploiting this valuable feature of a scene, which complements and enriches the geometric description that can be obtained by some of the methods above, *scene*

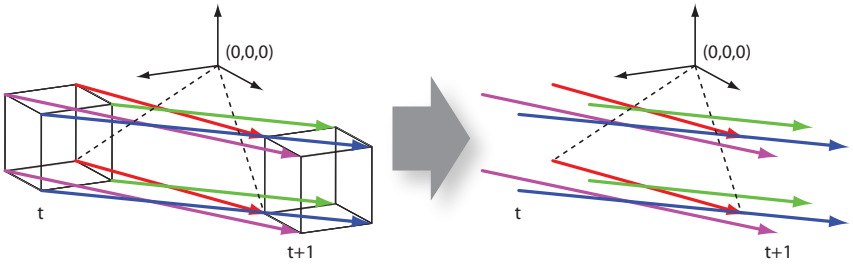


Figure 2.4: Scene flow. Each vector represents the motion of a feature of the reconstructed shape from one time instant to the next one. Note that the reconstructed shape has rotated 90 degrees around its center and also translated. The scene flow implicitly describes both types of motion

flow and *surface tracking* approaches can be found in the literature, which present important differences on the nature of the data used for estimation.

2.2.1 Scene flow

Earlier approaches have separated the shape and motion estimation tasks, in a *scene flow* formulation of the problem. The scene flow concept was introduced as a 3D equivalent to the *optical flow* in monocular video sequences. In this case, input data consist in sequences of images captured by a multi-camera setting.

The techniques in the literature first pre-compute shape estimates at each time instant. Then, velocity fields are obtained by feature matching between reconstructions at different instants. The scene flow typically complements photo hulls or multi-view stereo surface reconstructions. These techniques are reviewed next. Fig. 2.4 shows how rigid motion can be represented accompanying an existing shape at different time instants in a toy example.

As seen in [Vedula et al., 1999], when many cameras are used to image a scene, it is necessary to apply some regularization or smoothing to compute the scene flow describing the non-rigid motion of the scene. Reviewing the literature, we find two different ways to compute the scene flow for non-rigid motion, regarding the way regularization is applied:

- **Image-based regularization**, as in [Vedula et al., 2002]. This approach is used with voxel coloring for obtaining the voxelized photo hull of the scene and its scene flow. In this case, smoothing is performed in each image plane and the smoothed 2D optical flows are combined for obtaining the 3D scene flow.

- **3D regularization** on 3D surfaces. In [Carceroni and Kutulakos, 2001], this approach is used for recovering the 3D shape, reflectance and non-rigid motion of scenes in a surfel-based representation, with the scene flow smoothing

directly computed in 3D space. One of the most interesting points of this approach is the usage of a model for the appearance of the projections of surfaces in the scene not assuming Lambertian surfaces, although it requires the knowledge of the physical position of light sources.

Some works in the literature obtaining dense non-rigid scene motion estimates using a polygon mesh representation of surfaces, in a scene flow formulation of the motion estimation problem, can be found in [Ahmed et al., 2008] and [Klie et al., 2007]. The former receives as input a sequence of visual hull surfaces obtained from static surface reconstructions in each time instant. Then, sparse correspondences from SIFT features [Lowe, 2004] between adjacent frames are obtained. Finally, dense correspondences are inferred by finding matches at neighborhoods of each found correspondence. The latter [Klie et al., 2007] receives the same type of input and obtain a sparse 3D motion field from combining 2D optical flows from different views. Similarly to the previous case, sparse motion vectors are propagated to their vicinities in order to obtain a dense motion field. Dues to the noisy motion vectors at the output of the algorithm, temporal smoothing is applied in the form of low pass filters.

In [Starck and Hilton, 2005], a spherical model is constructed by mapping the surface obtained at each time instant to a sphere. In this case, surfaces are estimated by first concatenating voxel coloring and marching cubes [Lorensen and Cline, 1987], and then applying projective texturing [Mueller et al., 2004] in order to provide texture to the surfaces. As its authors claim, the spherical domain gives a consistent 2D parametrization of non-rigid surfaces for matching, which can be used in order to find the 2D bijection that guarantees a continuous one-to-one surface correspondence.

2.2.2 Surface tracking

More recent approaches exploit both spatial and temporal redundancies in sequences with motion by tracking surfaces from multi-view captures. Some of the existing techniques exploit multi-view data in a manner similar to multi-view stereo reconstruction, while others aim at obtaining shapes equivalent to photo hulls with wider-baseline scenarios in mind. Also for the latter scenario, silhouette-based approaches can be used to introduce precise deformations to an existing model in order to fit it to the silhouettes at each time instant. Other approaches aim at tracking objects in crowded scenes with very sparse cameras without color calibration. All these approaches are viewed in more detail next.

Multi-view stereo-based. Impressive time-coherent reconstructions can be obtained by tracking surfaces reconstructed by multi-view stereo, at the cost of large computation times. In [Furukawa and Ponce, 2008, Furukawa and Ponce, 2009b], normalized cross-correlation is used as a photo-consistency measurement between local neighborhoods of vertices in the mesh surface being tracked and images captured at the next time instant with a detailed treatment of visibility. First, smoothing constraints are applied at a local scale, in order to capture both radial and tangential surface motion. Then, at a global scale, a second level of refinement is achieved by introducing both smoothing and rigidity constraints.

In [Courchay et al., 2009], a global variational approach is applied in order to jointly estimate shape and motion. Their method fixes the topology of the surface, represented as a mesh, at the initial instant and then recovers both shape and motion by optimizing the positions of vertices at each instant. The optimization is driven by an energy function that incorporates photo-consistency and smoothness terms for both the velocity field and the surface across time.

In [Goldluecke and Magnor, 2004], *spatio-temporal* 3D hyper-surfaces are introduced in an elegant notation able to combine data from several time instants in a sequence in order to provide smooth surface estimates over time. The dynamic geometry is volumetrically modeled as a 3D iso-surface in space-time. To create the model, the photo-consistency of the whole scene is optimized with temporal smoothness constraints, which are intrinsic to the *PDE*-based evolution method used for energy minimization. The intersection of the 3D hyper-surface with planes of constant time gives the 2D surface at each time instant.

Silhouette-based. In [de Aguiar et al., 2007], range data is used for creating a high resolution mesh model of an object to be tracked. Then, *Laplacian* deformations are applied in order to fit the surface of the model being tracked to the set of silhouettes in each time instant, obtaining impressive offline results. In its follow-up [de Aguiar et al., 2008], photo-consistency constraints from multi-view stereo are introduced in the framework in order to provide accurate mesh fitting to small scale shape detail.

In [Guan et al., 2008], color information is introduced in a way that can be used in very wide-baseline scenarios and without requiring color calibration between cameras. A *Gaussian Mixture Model* (GMM) model of each subject in the scene is automatically learned in each view when a new shape is found in the scene. A *Bayesian* method is used to infer the voxel occupancy, combining the silhouette information with the matching score to each multi-view GMM model. This approach does not provide dense motion fields, but it can be useful for tracking multiple objects in crowded scenes with sparse multi-view settings.

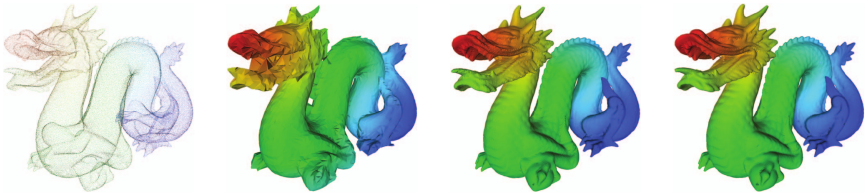


Figure 2.5: Meshing of a set of points. From left to right, an initial set of points and different surfaces represented as polygon meshes, obtained with increasing spatial resolution. Courtesy of Xavier Suau, extracted from [Suau et al., 2010]

2.3 Meshing Algorithms

Meshing algorithms provide a means to compute a closed, continuous surface out of a set of input points. Typically, the output surface is provided as a set of connected triangular facets. In Fig. 2.5, the data points from the *dragon* dataset in [The Stanford 3D Scanning Repository, 2010] are used to generate three different surfaces with increasing spatial resolution.

Regarding the literature in meshing algorithms from a set of oriented points, two families of methods can be considered. The first one comprises methods based on the propagation of an existing surface, whereas the second family comprises methods based on a volumetric reconstruction followed by marching cubes [Lorensen and Cline, 1987].

2.3.1 Propagation-based methods

Propagation methods for surface reconstruction have in common the initial definition of one or more regions that are used as seeds for a propagation algorithm that attempts to cover the whole surface between the input 3D points. Some of the methods available in the literature that fall in this category are:

- *Ball pivoting* [Bernardini et al., 1999]. It starts with a seed triangle and iteratively pivots a ball around an edge (*i.e.* it revolves around the edge while keeping in contact with edgepoints) until it touches another point, forming another triangle.

- *Restricted and oriented propagation* [Suau et al., 2010]. It starts by voxelizing the space where the surface lies and fills each voxel with an original point. Then, a specifically-designed propagation scheme connects close points in order to create new faces of the resulting surface.

Ball pivoting

The ball pivoting algorithm is an advancing-front algorithm to incrementally build an interpolating triangulation of a given point cloud. The method is conceptually simple. Starting with a seed triangle, it pivots a ball around each edge on the current mesh boundary until a new point is hit by the ball. The edge and point define a new triangle, which is added to the mesh, and the algorithm iteratively considers a new boundary edge for pivoting.

Restricted and oriented propagation

Restricted and oriented propagation is a method to reconstruct meshed surfaces from point sets based on a propagation through a voxelized space performed in order to find the closest neighbors of every data point. Propagation is done following a specific propagation pattern which exploits reciprocity in neighbor finding. Every obtained pair of neighbors defines a new edge of the output mesh.

2.3.2 Marching cubes-based methods

Marching cubes (MC) is a well-known method for extracting a polygonal mesh of an iso-surface from a 3D scalar field –regular voxelization–. A set of oriented points belonging to the surface of the objects –explicit surface representation– is a very common output of 3D reconstruction methods (the first part of [Furukawa and Ponce, 2007] constitutes an impressing multi-view stereo example.) It is possible to translate it to a volumetric representation, usually under certain conditions of spatial uniformity, in order to apply marching cubes at the last stage.

Therefore, some approaches exist which, first, determine voxel occupancy by means of a certain type of regularization and, then, extract the surface by means of MC. Some methods in the literature are:

- *Robust implicit moving least squares-marching cubes* [Oztireli et al., 2009]. It extracts the iso-surface of a *Moving Least Squares* surface, defined by a point set as a robust implicit extension of Moving Least Squares that preserves sharp features, by using non-linear regression and finally applies the marching cubes algorithm.

- *Poisson reconstruction* [Kazhdan et al., 2006]. It obtains volumetric occupancy by solving a Poisson equation and then performs marching cubes in order to extract the surface.

Robust implicit moving least squares-marching cubes

Moving least squares is a very attractive tool to design effective meshless surface representations. However, as long as approximations are performed in a least square sense, the resulting definitions remain sensitive to outliers, and smooth-out small or sharp features. In [Oztireli et al., 2009], these major issues are considered and a novel point-based surface definition combining the simplicity of implicit MLS surfaces [Kolluri, 2005] with the strength of robust statistics is presented.

To reach this new definition, MLS surfaces are reviewed in terms of local kernel regression, opening the doors to the utilization of a robust kernel regression. This representation, which can handle sparse sampling, generates a continuous surface, which better preserves fine details, and can naturally handle any kind of sharp features with controllable sharpness. It combines ease of implementation with performance competing with other non-robust approaches. At the latest stage, a triangular mesh is extracted by using the marching cubes algorithm.

Poisson Reconstruction

Poisson reconstruction proposes a formulation that considers all the points at once, without resorting to heuristic spatial partitioning or blending, and is therefore highly resilient to data noise. Unlike radial basis function-based schemes [Turk and O'Brien, 2005], the Poisson approach presented in [Kazhdan et al., 2006] allows a hierarchy of locally supported basis functions, and therefore the solution reduces to a well conditioned sparse linear system. A spatially adaptive multi-scale algorithm is used, whose time and space complexities are proportional to the size of the reconstructed model.

Using this approach, surface reconstruction is expressed as a Poisson problem, which seeks the indicator function that best agrees with a set of possibly noisy, non-uniform observations. This approach can, for example, robustly recover fine detail from noisy real-world scans. At its last stage, an octree-based marching cubes implementation extracts a triangular mesh from the volumetric representation.

2.4 Conclusions

The main techniques and representation formats for 3D reconstruction from a set of calibrated input views found in the literature have been introduced. The state of the art provides a number of methods, based on different approaches, that are suitable for different types of scenarios that can be considered when working with multi-view camera settings. A taxonomy of the methods and types of representa-

tion that translate relevant features from the multi-view input data to 3D features (occupancy, surface, color...) has been presented.

When focusing on the 3D reconstruction of the objects in a scene captured by a set of multiple cameras, which are useful for both representation and further processing purposes, different methods yield results with dramatic differences, regarding the nature of the data used for reconstruction and the requirements of the specific application. Thus, methods offering high detail at its output are costly and oriented to small-baseline scenarios, such as the multi-view stereo approaches, whereas methods which provide lesser detail are suitable to interactive applications demanding small processing times and can have an arbitrarily wide-baseline configuration, which influences the reachable level of accuracy on the reconstruction.

We can observe that the only visible 3D features in multi-view data are the projections of the surfaces –*i.e.* the interfaces between the interior of objects and air– of the objects contained in the actual 3D scene. Therefore, we propose to exploit surface representations, with the use of surface patches or polygon meshes, in order to efficiently represent the reconstructed 3D features, instead of using volumetric representations, which in general spend valuable computational resources in computing and storing the occupancy of empty (air) or invisible (interior of objects) regions.

In order to efficiently compute surfaces in multi-view video sequences, the application of a tracking strategy seems a suitable option. In contrast with techniques for motion estimation by computation of the scene flow, tracking strategies can be used to exploit the temporal correlation of the scene in order to either reduce the computation time or improve the accuracy and consistency of the surface. Since reconstruction techniques are in general costly, it will be important to provide tracking techniques for the reconstruction of sequences, which allow for a combined exploitation of spatial and temporal correlations in the position of the surfaces.

Polygon meshes appear as good candidates for representing the visible features in 3D scenes, due to their powerful post-processing capabilities, being easy to handle in standard hardware-accelerated graphics pipelines and providing an explicit description of the surface topology. Meshes can be obtained from a discrete representation, based on surface samples, by different techniques, as shown in Section 2.3. However, the reviewed techniques are either computationally demanding or inaccurate in terms of topological correctness. Therefore, it is also proposed to obtain a method able to obtain a topologically correct continuous surface with a low computational cost.

In order to summarize the challenges of this thesis already mentioned in the previous chapter related to the state of the art, we attempt to provide a framework for the efficient processing of the available data in multi-camera scenarios by obtaining

a single instance representation of the objects in the scene that:

1. Has little computational cost when compared to other methods delivering a similar accuracy in their results.
2. It is suitable for the reconstruction of multi-view sequences, exploiting both spatial and temporal correlations.
3. Contains all the relevant information contained in the original images.

Finally, it is important to remark that, although this thesis is not focused on specific higher-level computer vision or computer graphics problems such as pose estimation or techniques for scene visualization –for free-viewpoint video–, the ultimate goal of the obtained scene representation is to be used by these applications, without the expense of operating directly on the data captured by a multi-camera setting that would arise in image-based approaches.

Chapter 3

Approach

The previous chapter shows that the availability of multi-view data is beneficial in order to provide state-of-the-art performance in several disciplines in computer vision. In addition, the rich multi-view information can be more easily exploited when presented as a *single instance* of a 3D data structure properly describing all features of interest. Imposing the condition that the reconstruction is accurate, with a minimal information loss, it is always possible to go back to a good estimate of the original images by re-projecting the 3D models onto the original viewpoints. The methodology for obtaining this representation must solve early vision concepts –such as visibility, occlusion or image distortion– and leave for a posterior stage the feature analysis for specific applications.

One of the main goals of this approach, which can be interpreted as a low-level data fusion from multiple sensors, is to avoid using a *multi-instance* representation composed by the raw images from each available viewpoint and their corresponding calibration data at an application level. Such a multi-instance representation would further impact an analysis application attempting to exploit not only multi-view data, but also motion information by processing multi-view video sequences. Such scenario might require costly processing, transmission and storage of multi-view data from several time instants. Furthermore, algorithms based on the processing of a single instance 3D representation of a multi-view scene can be more general, because they will never depend on the number of available cameras and their interactions. Otherwise, the cost of processing multi-view data could exponentially grow with the number of input views.

This thesis attempts to demonstrate that surface representations of the elements of interest in multi-view video sequences can be obtained at a reasonable computational cost, providing complete and compact descriptions of the relevant visual features available in the original multi-view streams. In other words, this type of

description can provide enough visual information to analysis applications as to render image-based approaches unpractical, due to the additional management cost associated to using raw multi-view data in the latter.

This chapter describes the approach presented throughout this thesis, designed to provide a complete and compact multi-view data representation of foreground elements in dynamic scenes, which can be used by both analysis and visualization applications. First, the set of visual features included in the proposed surface representations are presented and classified regarding their nature. Then, the necessary methods to obtain each of these representations are introduced. Finally, the organization of the rest of the dissertation is presented.

3.1 Surface Description

Relevant visual features in multi-view scenes are always found on the surfaces of objects of interest. Indeed, the interior space of objects of interest is typically hidden by an opaque surface, whereas empty space occupied by air does not provide any visual information. Therefore, it is this interface between the interior of objects and empty space the one that contains the relevant visual data.

Surfaces in 3D space can be represented in a number of ways, either explicit or implicit –a volumetric field determining spatial occupancy implicitly contains the location of surfaces–. In our approach, surfaces are explicitly described by means of two different, although related, types of primitives. The first representation consists in a sample-based description of the surface, whereas the second one is a polygon mesh.

One important advantage of considering a surface sampling scheme instead of a volumetric approach is that it is possible to augment the level of detail exactly on the surface of interest, by increasing the sampling density in that region. In contrast, a volumetric approach would require an increase of the sampling density in a region which would comprise the surface, part of the interior of the object and part of the empty space around the object, due to the impossibility of exactly determining the position of the surface.

The benefit of a polygon mesh compared to a sample based description is that it provides a mechanism –linear interpolation– to determine an estimate of the surface between existing samples. In comparison with other possible interpolation schemes, this one is supported by current mainstream graphics hardware, which makes it a good choice for interactive applications.

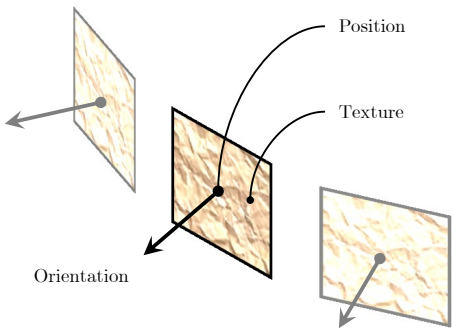


Figure 3.1: Three surface samples of the proposed 3D representation characterized by their geometric (3D position and orientation) and texture descriptors. The size of each surface sample in the proposed 3D representation is considered to be infinitesimal

3.1.1 Sample-based surface description

The first type of support for the description of 3D objects of interest in a multi-view scene is, as mentioned, a collection of surface samples. In its most elementary form, such representation consists in a point cloud with the 3D positions of a certain number of surface points. However, with the help of additional descriptors, higher degrees of information detail can be obtained.

In order to describe the shape of a 3D surface as estimated from data captured by a calibrated multi-camera setting, geometry and texture are complementary features which, combined, provide a rather complete set of the available visual features. Whereas the former describes the position and pose of a 3D surface element, as referred to a reference coordinate system, the latter describes its color at a local scale. An illustration of the concept of a surface sample is shown in Fig. 3.1.

Geometric descriptors

Geometric information can be expressed in different manners, depending on the requirements of the application, ranging from very low level descriptions, which require less computational resources for their extraction and representation, up to higher level descriptors, the extraction and representation of which require more resources.

In general, three different types of geometric descriptors can be considered for each surface sample:

- *3D position* $\mathbf{x}_i$, with respect to an arbitrary 3D reference system.
- *Orientation* $\hat{\mathbf{n}}_i$, a unitary outward-pointing vector in a direction normal to

the surface.

- *Curvature*, described by its two principal curvatures $\mathbf{k}_{1,i}$ and $\mathbf{k}_{2,i}$ [Guggenheimer, 1963].

At this point, it is interesting to discuss the properties of these three descriptors. The 3D position of a surface sample constitutes the most basic geometric description. By considering the variation of the position of the points included in a conveniently sized neighborhood of the surface sample, a higher level descriptor –its orientation– can be estimated. At the highest level, the variation of the orientation of the points belonging to the neighborhood of the surface sample can be used to estimate its curvature parameters. However, this third type of descriptor will not be used in our approach, for curvature can be determined at any later stage by exploiting the extracted location and orientation of a set of infinitesimal or *elementary* surface samples at close distances.

Alternative approaches consider larger surface samples [Carceroni and Kutulakos, 2002], which require more detail in the represented features (curvature, orientation and also texture). In this case, even the computation of the orientation of each surface sample would require more than the knowledge of the location of a set of close surface samples and a hint about the interior and exterior of the object, because samples would be too distant to describe local variations. Thus, an interesting property of elementary surface samples is that, given a sufficient sampling density, a small amount of information at every point –location and orientation– suffices to geometrically determine the surface.

Texture descriptors

Texture can also be encoded as an attribute of each surface sample. Assigning an RGB color that is consistent with each one of the viewpoints that have a line of sight with the surface point described by the sample already constitutes a low-level texture description. Such a method for describing texture can have an arbitrary level of detail, by varying the density of the surface samples.

Higher-level features, such as *Gray-Level Co-occurrence Matrix* (GLCM)-derived figures [Haralick et al., 1973] or edge histograms can also be used when a low density of surface points is needed for implementation constraints. The use of this type of features allows using a sparser surface description while keeping valuable information about the texture of the object. Therefore, they can constitute a useful complement to the low-level RGB color descriptor when the sampling density over the surface cannot provide enough level of detail of its texture variations, which are in general subtler than their geometric counterparts.

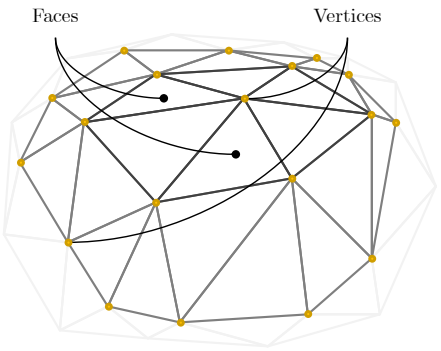


Figure 3.2: Local detail of a polygon mesh describing a surface. Each triangular face, which might contain texture information, shares its vertices with other neighbor faces, thus describing the surface topology

In this thesis, only the RGB color has been used as a texture descriptor, assuming a sufficient surface sampling density. Again, the adoption of an approach based on elementary surface samples, instead of larger surface samples, results in a benefit in terms of the required per-sample information. For both types of features, geometric and texture, a rather complete description of the actual surfaces can be obtained by encoding a small amount of data (only position, orientation and color) for every sample when a sufficient sampling density is considered.

3.1.2 Polygon mesh surface description

In order to describe the reconstructed surface even in regions lying between elementary surface samples, a polygon mesh is used. Polygon meshes can be considered a first-order polynomial interpolant –or *linear* interpolant– of the actual surface, given a set of elementary surface samples. It constitutes, therefore, a suitable continuous representation of the information contained in a set of discrete surface samples.

There are different ways to describe a polygon mesh, such as lists of vertices and faces or a half-edge structure. In our case, we consider a the former type of description, composed by triangular faces that share a set of vertices. An illustration of such description is shown in Fig. 3.2. However, the latter is also used at an intermediate step in order to check the topological correctness of the polygon mesh.

Some properties regarding the topology of a mesh must be fulfilled if the mesh is used for analysis. A two-manifold mesh can be properly oriented, whereas a mesh not fulfilling this topological property cannot be, for example, unfolded onto a 2D plane [Floater and Hormann, 2001]. Since surfaces reconstructed from multi-view video sequences using the proposed approach are going to be used for both visualization and analysis, special care will be taken in order to provide surfaces

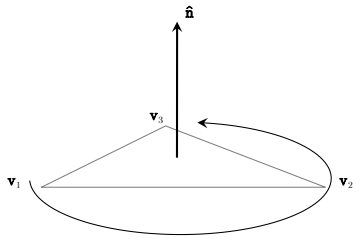


Figure 3.3: Ordering of the three vertices of a triangular face following the right-hand rule, given the orientation of the face

with correct topology.

Geometric descriptors

As pointed out above, surface geometry described by polygon meshes is considered to be consisting of triangular faces that share their vertices with neighbor faces.

Vertices. Similar to the case of surface samples, vertices in a mesh can contain different types of geometric descriptors. Apart from the 3D position of the vertex in an arbitrary reference system, a commonly included feature is an orientation, which can be used to compute surface illumination. However, unless otherwise stated, only the 3D position of each vertex will be considered as a per-vertex feature throughout this dissertation.

Faces. Typically, triangular faces are geometrically described as a set of three indices to their corresponding vertices. This is due to the fact that their orientation is commonly assumed to follow the right-hand rule of ordering of their vertices, as shown in Fig. 3.3.

Texture descriptors

Without considering more sophisticated effects such as *mip-mapping* –used in computer graphics in order to provide superficial detail beyond the geometric resolution of the polygon mesh–, two types of texture descriptors can be considered, regarding a polygon mesh (geometrically) described by means of lists of vertices and faces.

- *Face texture*, containing high resolution detail between vertices.
- *Per-vertex RGB color*, used to interpolate the surface color between vertices.

Due to the minimalistic texture representation chosen in surface sampling, the most natural choice for the mesh representation is a per-vertex color description. Such a texture description will be accurate as long as the vertex density of the mesh is sufficiently high. Again, the choice of an elementary surface sampling approach with a sufficient density allows encoding the continuous surface with a small amount of data when compared to that of providing a texture to each face composing the mesh. Indeed, in a closed surface with N_f faces, approximately $N_v = 2N_f$ vertices exist. Whereas encoding the texture coordinates in a parameterizable two-manifold mesh would require N_v 2D coordinates –$3N_f$ in non-manifold surfaces– and a texture image for each surface only N_v RGB colors (3D coordinates) are required for per-vertex color information.

In this section, two types of surface representation have been presented, which will be used at different stages throughout the methodology presented in this thesis. Such methodology, resulting in the extraction of these two types of surface descriptions, is introduced in the next section.

3.2 Methodology

Several factors strongly influence the design decisions in this thesis towards methods able to work with arbitrarily wide baselines in real-time scenarios. Examples of these factors are the computational cost of processing an arbitrarily high number of cameras or the impossibility of placing a desired number of cameras due to spatial and usability constraints.

Whereas multi-view stereo methods [Campbell et al., 2008, Esteban and Schmitt, 2003] provide state-of-the-art quality with properly conditioned small-baseline set-ups, the surface reconstruction methods proposed in this thesis might fall one step behind those in terms of quality in properly-conditioned scenarios. However, this strategy allows us to provide surface reconstruction in challenging wide-baseline scenarios, where more accurate methods fail due to the absence of the required availability of redundant data.

The two different types of surface representations that are considered in the following chapters are used in two different stages of the processing chain depicted in Fig. 3.4. Therefore, sample-based representations of the reconstructed surfaces are obtained at the end of the *surface sampling* stage, whereas continuous polygon mesh representations are available after the *surface interpolation* stage. Due to the different nature of these two methods and type of representation, this thesis has been divided in two parts, surface sampling and surface interpolation, tackling the specific challenges at each of these stages. These are described next.

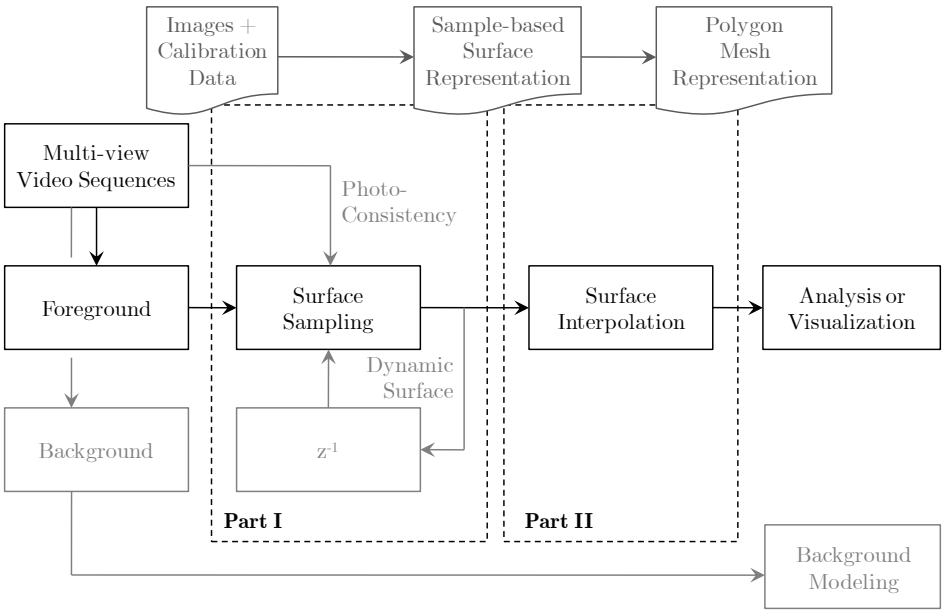


Figure 3.4: 3D surface reconstruction as a two-stage approach. After the first stage, a sample-based representation of surfaces is obtained, whereas, after the second one, a polygon mesh is obtained as the interpolation between the surface samples.

3.2.1 Surface sampling

The surface sampling methods presented in this thesis can be classified by two related criteria:

- The type of *available data*, leading to different possible reconstruction methods, some of them faster and others more accurate and offering richer information of the surfaces.
- The type of *search strategy* for each surface sample, leading to different types of algorithms, some better adapted to exploit photo-consistency and others to exploit temporal correlation.

In the following, the differences that appear from the choice of one of the types of available data and search strategies are briefly introduced.

Reconstruction methods and available data

As depicted in Fig. 3.4 Part I, a number of possible pipelines has been considered in order to extract surface samples from multi-view data. Thus, in case of considering only the silhouettes of objects of interest for a single time instant, a first type of surface reconstruction (*silhouette-consistent*) appears. This initial pipeline can

be slightly modified to exploit temporal correlations (*dynamic surface*) or color information (*photo-consistency*).

Silhouette-consistent reconstruction. The reconstructed surfaces obtained by this approach are those of a *visual hull* [Laurentini, 1995, Franco and Boyer, 2009], corresponding to the techniques in Section 2.1.2 in the previous chapter.

Photo-consistent reconstruction. When attempting to obtain a single-instance representation containing all the relevant features of the input images, it is necessary to also consider color or texture information. As seen in Section 2.1.4, it is useful to exploit both binary silhouettes of the objects of interest and color images [Kolev and Cremers, 2008]. The former data provides a strong robustness and the ability to reconstruct shapes in multi-view scenarios with an arbitrarily wide baseline by limiting the spatial search area. The latter, color information, allows refining shapes by enforcing photo-consistency between images, using techniques similar to those in Section 2.1.3.

With this approach, apart from obtaining tighter surface estimates than those obtained with the silhouette-consistent approach, additional shape features are also directly available at the output: a texture descriptor –such as a RGB color– is included at each surface sample when photo-consistency is also enforced. Please note that a texture descriptor could also be attached to silhouette-consistent reconstructions, although photometric information would not be used to improve the accuracy of the reconstructed surface. In turn, this latter approach involves a smaller computational cost.

Dynamic surface reconstruction. Inter-frame correlation is exploited in this thesis for reducing the computational cost involved in the reconstruction of surfaces in consecutive time instants of multi-view video sequences. Interestingly, as presented in Section 2.2.2, the exploitation of the correlation of a scene between consecutive time instants can also provide an estimate of the motion of each surface sample.

Reconstruction methods and search strategies

Surface sampling can be intuitively interpreted as the distribution of samples, at a constant or varying density, all over the surfaces of the objects that are being reconstructed. One desirable property related to the distribution of samples would be that their density is increased where required, while keeping a lower density in areas without relevant information.

Whereas this would be difficult to obtain with classical volumetric techniques, where 3D space is arbitrarily divided in order to determine the occupancy of each subdivision –with some possible refinements such as *octrees* [Szeliski, 1993]–, the surface reconstruction methods present here an advantage, for they are free of these arbitrary divisions and sampling is more suitable to be adapted to the contents of the scene.

Thus, a *search* strategy specifically designed to exploit the features of the available data are expected to result in the placement of surface samples in valid positions.

As it will be seen in subsequent chapters, the choice of a strategy for search of surface samples leads to different algorithms. Some of these will present convenient behaviors to exploit temporal correlation, whereas others will adapt better to the exploitation of photo-consistency. A possible classification of the methods presented in this thesis is:

- *Image-based* search, with search regions defined along the back-projected rays of camera pixels, where surface samples are associated to individual pixels.
- *Surface deformation-based* search, with search regions along the normals of each surface sample, assuming a certain level of correlation between the surfaces in consecutive time instants.
- *Statistical* search, with search regions distributed around other valid surface samples, where the likelihood of finding new surface samples is higher.

The advantages and disadvantages of these methods will be presented and discussed throughout the chapters dedicated to surface sampling.

3.2.2 Surface interpolation

Surface interpolation corresponds to the *Part II* in Fig. 3.4. The input data consists of surface samples with or without color information. Although there exist several methods in the literature for extracting a polygon mesh from a set of oriented surface samples [Bernardini et al., 1999, Kazhdan et al., 2006], the target of the technique proposed here is to obtain a high degree of precision while keeping a reasonably reduced computational cost.

Some methods in the literature [Kazhdan et al., 2006] consist in first robustly determining the volumetric occupancy of 3D space and then extracting the surface as the interface between occupied and empty space (Section 2.3.2). The method proposed in this thesis shares similarities with another family of mesh extraction methods, which can be referred to as propagation-based schemes (Section 2.3.1),

with [Bernardini et al., 1999] constituting a well-known example. The main advantage of our method with respect to other propagation-based schemes is the usage of fewer computational resources, accomplished by the use of suitable data structures and a quick propagation algorithm based on a set of rules for guaranteeing the desired topological properties and a high accuracy.

At the output of this stage, an interpolating surface is obtained from the set of input surface samples. This surface can be used for both visualization and analysis, thanks to its topological correctness.

3.3 Thesis Organization

The main body of this thesis dissertation is divided in three parts. The first one presents a set of methods suitable for surface sampling from different types of input data. The second part describes the proposed meshing algorithm for surface interpolation. The third part presents the global results combining methods from Parts I and II, as well as some concluding remarks and proposed lines of future work.

3.3.1 Part I

The first part of the document contains chapters dedicated to the different surface sampling schemes, providing some advantages and disadvantages for each data configuration considered for reconstruction. In the final one, the techniques are validated and compared. The information presented in each of these chapters is the following:

- In Chapter 4, an *image-based surface sampling* approach with search regions consisting in the back-projected rays of each pixel, exploiting photo-consistency.

- In Chapter 5, a surface sampling based on the *deformation* of an existing surface, with search regions along the normal of each surface sample, exploiting temporal correlation.

- In Chapter 6, a surface sampling scheme based on the *statistical propagation* of surface samples in regions with higher likelihood of containing surfaces, exploiting spatial correlation.

- In Chapter 7, experiments with each one of the proposed techniques and the corresponding results, followed by a comparison of the three methods.

3.3.2 Part II

The second part of the dissertation contains two chapters dedicated to the extraction of a mesh interpolating a continuous surface between surface samples. First, the methodology is introduced and then, in Chapter 9, results are presented. The contents of these chapters are:

- In Chapter 8, a *surface interpolation* approach –meshing algorithm– is presented, which provides accurate polygon meshes with a low associated computational cost.
- In Chapter 9, some results obtained with the interpolation technique are presented.

3.3.3 Part III

The two previous parts of the thesis include partial results corresponding to the sampling and interpolation techniques. The third part of the thesis dissertation contains overall results, validating the complete approach –sampling and interpolation– and the conclusions. The chapters conforming this last part of the document contain:

- Chapter 10 presents quantitative and qualitative evidence of the suitability of the complete approach –surface sampling and interpolation– for the problem of surface reconstruction.
- Chapter 11 lists an overview of the accomplished goals and proposes some lines of future work.

Finally, Appendices A, B, C and D introduce, respectively, concepts related to the geometry of calibrated cameras, the *kd*-tree structure for efficient spatial queries, the *Hausdorff*-distance as a metric for surface quality and a list of publications obtained as a result of the research conducted in the elaboration of this thesis.

Part I

Surface Sampling

Surface Sampling

This part of the dissertation introduces a methodology for providing a complete, discrete representation of the surfaces of moving objects –scene foreground– that are observed by a rig of calibrated cameras. This is accomplished by a set of procedures oriented to the finding of the location and orientation of surface samples by means of specifically designed search schemes aiming at efficiently exploiting the spatial or temporal redundancies available in the input multi-view data.

Surface sampling, when compared to other 3D reconstruction methods, provides a higher degree of freedom in the desired output resolution –in this case, sampling density–. Although this might difficult the process of interpolating between surface samples, the technique presented in Part II correctly handles this possible limitation. In volumetric analysis, computational costs grow when an increase of the sampling density at the visible features of the 3D scene –surfaces of objects– is desired, due to the uniform distribution of samples in the entire working volume. However, in surface-oriented approaches, sampling density can be more freely increased, because the additional computational cost will only be used for samples representing objects surfaces.

In the following chapters, three different techniques are presented, which develop the principle of finding the position and orientation of surface samples in different manners. Regarding different types of input data –binary silhouettes or color images, multi-view video sequences or static multi-view scenes–, each of the proposed techniques shows some advantages and some limitations. These techniques are the result of an evolutionary approach in the search for the best possible technique that efficiently exploits the available information and improves the limitations of each previous one.

Chapter 4

Image-based Surface Sampling

As presented in Section 2.1.1, some approaches for scene reconstruction from a set of calibrated views show a high level of robustness, such as the silhouette-based methods, whereas others attempt to obtain the finest possible detail, such as current multi-view stereo techniques. Photo-consistent techniques fill in the gap between these two types of approaches, providing photo-hulls. In certain environments, it is required to obtain a complete and robust representation of a multi-view scene under tight constraints on the smallest achievable baseline of the camera setting. In such a scenario, photo-consistency can provide better accuracy with respect to the real shapes than silhouette-based methods, while being more flexible in terms of baseline than multi-view stereo approaches.

In order to exploit photo-consistency, an image-based approach appears as a good candidate to reconstruct surfaces from multi-view data [Furukawa and Ponce, 2007]. Therefore, in this chapter an image-based sampling strategy is presented, which provides a photo-consistent surface reconstruction of foreground objects in scenes with known background.

4.1 Motivation

Multi-view stereo techniques aim at reconstructing the 3D shapes of objects from images captured from small baseline multi-view setups. This setup permits reconstructing surfaces of objects in high detail, as long as the surfaces are visible from a large enough number of viewpoints. When the volume enclosing the scene to be reconstructed is large, more viewpoints must be placed around the scene in order to

keep a small baseline. However, it is not always possible to arbitrarily increase the number of viewpoints, due to either physical –camera placement– or computational constraints.

The goal of the presented method is an alternative approach to tackle the problem of improving accuracy without greatly increasing the number of views. The target scenario consists in a scene in an indoor environment with known background whose images are captured from a highly sparse set of viewpoints, as few as *e.g.* 4 or 5 –as the cardinal points and a top view– in a wide baseline setup. These viewpoints are placed around the whole scene in a way that they offer a wide spatial coverage. In contrast with the usual *sparse* scenarios that are considered in multi-view stereo, which typically include between 10 and 20 viewpoints, the discrepancies due to perspective changes between images in such setups are dramatic and influence the reachable reconstruction accuracy.

4.1.1 Related work

The *visual hull* was introduced in [Laurentini, 1994] as the maximal 3D shape consistent with the silhouettes of an object projected onto a set of viewpoints. Several methods in the literature, usually referred to as *shape-from-silhouette*, estimate the visual hull. In these methods, input data consists solely of binary silhouettes. In [Franco and Boyer, 2009], a polygon mesh is directly reconstructed, which represents the surfaces of the visual hull. This technique provides a highly precise surface description when compared to most common volumetric counterparts.

A pre-processing stage, which may introduce segmentation errors, must provide accurate silhouettes of the objects of interest. In order to make visual hulls more robust against errors in silhouettes, [Franco and Boyer, 2005] and [Landabaso et al., 2006] provide algorithms for obtaining visual hulls from silhouettes presenting false detections and misses, both using a voxel grid representation.

The *photo hull* was introduced in [Kutulakos and Seitz, 1999] as the maximal 3D shape which is photo-consistent with the set of input images. In this case, the captured images are used as input data, instead of binary silhouettes. In their work, a voxel grid is again used as 3D support. The algorithm proposed for extracting the 3D shapes, called *space carving*, consists in setting all voxels of an initial volume that contains the actual scene as occupied and iteratively carving away those visible voxels that, visited using *sweep planes* –in order to avoid computing the visibility of each voxel from each camera–, do not pass a photo-consistency test. Due to the richer information available, the photo hull constitutes an estimate of the actual shape tighter than the visual hull, allowing concavities on the reconstructed shapes.

Voxels are useful to represent dense volumes, but they do not accurately describe

their surfaces. Alternatively, *surface samples* are designed for that purpose. Several examples in the multi-view stereo literature, like [Carceroni and Kutulakos, 2001] and [Furukawa and Ponce, 2007], have already used this type of representation, either to provide the final result or as an intermediate representation. In [Furukawa and Ponce, 2007] an initial sparse set of surface samples is obtained from matching image-based features. From this initial estimates, an iterative algorithm augments the density of samples and rejects outliers, achieving results with high accuracy. Then, a polygon mesh is morphed with an energy minimization algorithm until it fits the set of reconstructed surface samples.

In [Campbell et al., 2008], another common strategy is utilized, which consists in, first, estimating the depth map of each available view by using local groups of input images and then combining them in a voxel grid, in order to obtain the global surface estimate. Usually, combining methods use some regularization technique. Minimization via graph cuts offers fast performance when minimizing a certain domain of energy functions with binary variables, as defined in [Kolmogorov and Zabih, 2004].

In some other works, such as [Kolev and Cremers, 2008], the robustness of the visual hull is combined with color information, allowing the reconstruction of concavities. A convex functional is minimized with the exact silhouette consistency imposed as a constraint that enforces the resulting shape to fit in the binary silhouettes once re-projected onto the original viewpoints. However, multi-view stereo techniques are designed for scenarios where the robustness against the dramatic discrepancies between views present in wide-baseline setups is not a determining factor and, therefore, do not suit the requirements imposed by our target scenario.

4.1.2 Problem statement

The proposed method aims at reconstructing the photo-consistent surfaces of objects, the images of which have been captured by a very sparse set of cameras with known pose and intrinsic parameters. Similarly to [Furukawa and Ponce, 2007], the presented method exploits photo-consistency by performing an image-based search for surface samples. From [Kolev and Cremers, 2008], the proposed method keeps a resemblance in the sense of exploiting binary silhouettes in addition to photo-consistency measurements in order to constraint the position of the surface samples The complete method consists in the concatenation of three stages:

1. First, candidate depths for the image points in each view are obtained, based on photo-consistency measurements between pairs of pixels from neighbor cameras.

2. Then, the surface points are extracted by globally imposing photo-consistency

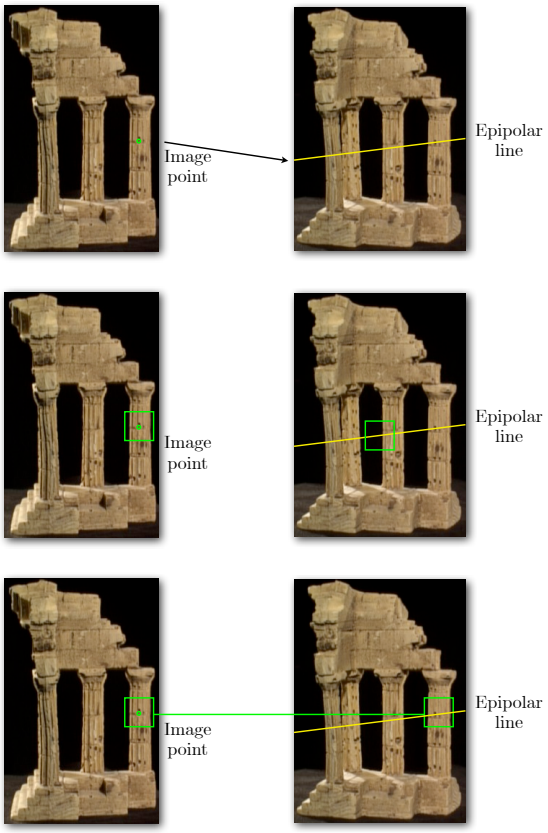


Figure 4.1: *Normalized Cross-Correlation* (NCC) exploits perspective similarity in order to find accurate matches between neighbor views when used in typical multi-view stereo scenarios. A match for a given image point is found along its epipolar line on the neighbor view

[Kutulakos and Seitz, 1999], considering the complete set of available views.

3. Finally, a normal for each surface point is estimated by processing the local geometry of its neighborhood.

4.2 Photo-consistency Test

Due to the severe perspective discrepancies between different views –wide-baseline setups–, the properties of photo-consistency measurements used in state-of-the-art multi-view stereo approaches do not match our specific requirements, despite of their great level of precision.

A widely utilized form of photo-consistency measurement in multi-view stereo, the so-called *normalized cross correlation* (NCC), assumes a certain perspective similarity between viewpoints, which actually exists in small-baseline image sets.

This assumption permits measuring the NCC in a straight-forward, fast manner by just computing it over 2D windows at each image, as illustrated in Fig. 4.1. In scenarios with a slightly wider baseline, a planar grid can be defined in 3D space, with sample points on the grid being projected onto the images. This grid must vary its orientation depending on the actual 3D geometry of the surfaces corresponding to the observed projections, as shown in [Furukawa and Ponce, 2007]. This alternative is computationally expensive, since both photo-consistency and orientation have to be jointly extracted and there remains the assumption of a maximal baseline that is too small for our target scenario.

In dynamic scenes where background images are available, foreground silhouettes of objects of interest in each image can be automatically extracted with the technique proposed in [Stauffer and Grimson, 1999]. Silhouettes are useful for photo-consistency for two reasons: on the one hand, they reduce the search space in the 3D domain; on the other hand, they can be used to reject many photo-consistent matches that may lie far from the actual object surfaces. Therefore, a simpler photo-consistency test with little impact from the perspective discrepancies can be considered if it is applied in combination with the constraints imposed by the silhouettes. In order to test the photo-consistency of a 3D point that belongs to the visual hull, we choose a less computationally expensive photo-consistency test that measures the squared Euclidean RGB distance between the pair of image points where it projects[1]. In order to cope with non-Lambertian surfaces, the photo-consistency measurement admits an additional tolerance to brightness differences when the chromaticity shows a variation below an adjustable threshold. A more detailed description of the photo-consistency test applied to pixels belonging to the projections of objects of interest follows.

A robust photo-consistency measure for non-Lambertian surfaces, which results in different projections of the surface showing different brightness levels, should account for the possible discrepancies in both their brightness and chromaticity components. Based in the foreground/background color-based segmentation scheme presented in [Horprasert et al., 1999, Xu et al., 2005], brightness distortion (BD) can be defined as a scalar value that brings expected photo-consistent pixels close to the observed chromaticity line. Color distortion (CD) can be defined as the orthogonal distance between the expected photo-consistent color and the observed chromaticity line. Both measures are shown in Fig. 4.2 and formulated in Eqs. 4.1.

$$\begin{aligned} BD &= \arg\min_{\alpha} \left(\mathbf{RGB}_1 - \alpha \cdot \mathbf{RGB}_2\right)^2 \\ CD &= \|\mathbf{RGB}_2 - BD \cdot \mathbf{RGB}_1\| \end{aligned} \tag{4.1}$$

[1] In our experiments, using other color spaces, such as YUV or LabCIE, did not provide clear improvements with respect to RGB.

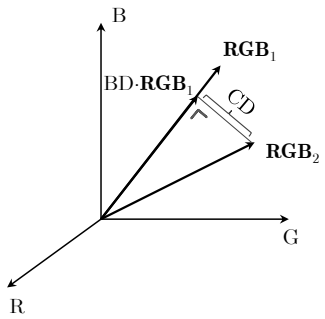


Figure 4.2: Distortion measurements in RGB space. $\mathbf{RGB}_1$ and $\mathbf{RGB}_2$ are the RGB values of two pixels from different images. CD is the chromaticity distortion between both colors and BD is the corresponding brightness distortion

In practice, BD can be easily computed as $BD = \mathbf{RGB}_2 \cdot \mathbf{RGB}_1 / \|\mathbf{RGB}_1\|^2$. A set of thresholds and rules are necessary to define the final photo-consistency test. This test, which takes a maximum RGB distance (RGB_{max}), has the following form:

1. If $\|\mathbf{RGB}_1 - \mathbf{RGB}_2\| < RGB_{max}$, then pixels are photo-consistent.

2. Else, if $\|\mathbf{RGB}_1 - \mathbf{RGB}_2\| \geq RGB_{max}$, but $CD < 10$ and $0.5 < BD < 1.25$, then pixels are also photo-consistent.

3. Otherwise, pixels are not photo-consistent.

This photo-consistency test, the thresholds of which have been empirically determined in order to obtain the desired behavior, provides a robustness improvement for photo-consistency measurements when considering non-Lambertian surfaces. In fact, when chromaticity remains similar enough between views, the range of allowed brightness distortion will include a wide range of shadowing and highlighting effects.

4.3 Sampling of Visible Photo-consistent Surfaces

Although the technique presented next could also be used to reconstruct the surfaces of a visual hull by using only binary silhouettes as input data, in the following we consider the more general configuration of using both color images and binary silhouettes of objects of interest. The method consists in generating lists of candidate locations for photo-consistent surface samples along the back-projected ray of each pixel and then imposing global photo-consistency by means of a carving procedure.

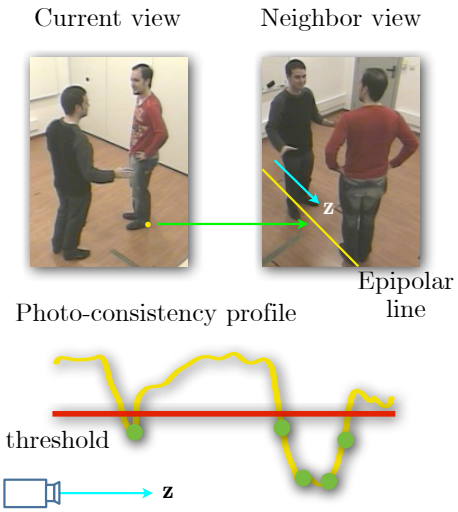


Figure 4.3: A list of photo-consistent candidates for a given pixel (marked in yellow in the left view) is obtained from testing pixels along its epipolar line in a neighbor view and stored in a list, in order of increasing depth with respect to the current viewpoint

4.3.1 Automatic neighborhood selection

In order to generate lists of photo-consistent candidates –that will contain the resulting surface points– from pixels in every view, a set of neighbor views for the image-based search must be defined.

The neighborhood of each viewpoint is automatically obtained by taking its closest viewpoints, defining the proximity of two viewpoints as

$$\pi_{ij} = \pi_{ji} = \left(1 - \frac{1}{2}\cos(\alpha_{ij})\right)\delta_{ij}, \tag{4.2}$$

where α_{ij} is the angle between the two unitary vectors orthogonal to the image planes of viewpoints i and j and δ_{ij} is the Euclidean distance between the centers of projections –Appendix A– of the two viewpoints.

4.3.2 Extraction of candidate surface points

For each viewpoint v_c of the complete set V, we define two sets of neighbor, N_c, and complementary, C_c, viewpoints, which have empty intersection and such that $N_c \cup C_c \cup v_c = V$.

Any image point back-projects to the 3D world as a ray with its origin at the center of projections of the given viewpoint, as seen in Appendix A. Thus, if we want to know the exact position of the 3D point which originated the captured 2D image point, we must search for it along the mentioned back-projected ray. In

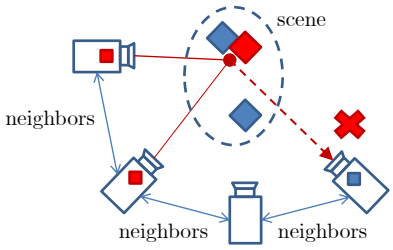


Figure 4.4: A candidate surface point visible but not photo-consistent in a third viewpoint is rejected. The colored square inside of each camera represents the color of the corresponding pixel when the 3D point is visible.

order to do so, we utilize the information available in the other views by means of a valuable geometric tool, the *epipolar line*, also introduced in Appendix A.

As illustrated in Fig. 4.3, for a given 2D point x_c that lies on the extracted foreground silhouette of an image i_c corresponding to the viewpoint v_c, we obtain the epipolar line $l_n(x_c)$ at each neighbor viewpoint $n \in N_c$. For each foreground pixel p traversed by $l_n(x_c)$, we measure its photo-consistency with the pixel at the point x_c. If the photo-consistency test defined above is positive, we triangulate the 3D position of the new candidate surface point. If the triangulated point projects to a foreground pixel for each view in $C_c \cup N_c$, we store it in a list of candidate 3D points $\{z_{x_c}\}$ with an increasing value of depth with respect to the center of projections of the view v_c.

At the end of this stage, a list of candidate surface points for each image point in each view is obtained. Furthermore, since the lists of candidate points are ordered in increasing depth from each viewpoint, estimates of the depth maps are implicitly obtained. The surface described by the points at the end of this stage can be interpreted as a shape which is already tighter to the actual shape than the visual hull provided by a *shape-from-silhouette* method, although it does not yet constitute a photo hull, since it lacks global photo-consistency.

4.3.3 Global photo-consistency

The next stage consists in discarding candidate surface points which are not photo-consistent with the complete set of viewpoints V. In order to do so, we will carve away those 3D points which do not pass the photo-consistency test defined in Section 4.2 for at least one of the viewpoints.

The algorithm works as follows: for each viewpoint v_c, we take the first point at every $\{z_{x_c}\}$. We consider the case that the 3D point obtained from two neighbor views is visible from another viewpoint, meaning that its distance to the center of

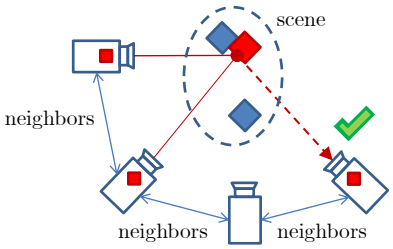


Figure 4.5: A candidate surface point visible and also photo-consistent in a third viewpoint is taken as the new closest candidate in this viewpoint. The colored square inside of each camera represents the color of the corresponding pixel when the 3D point is visible.

projections is smaller than the current depth estimate for the corresponding image point. If the candidate point is photo-inconsistent with the new viewpoint, it is removed and the next closest candidate for each view where it was visible is taken. This situation is illustrated in Figure 4.4. Otherwise, if the 3D point is photo-consistent with the new viewpoint, then we set it as the new closest candidate for the corresponding image point. This second situation is illustrated in Figure 4.5.

This filtering strategy is similar to space carving, with the difference that, instead of carving visible voxels, we carve candidate surface points along back-projected rays. As an advantage, the proposed method does not introduce additional spatial sampling artifacts due to the arbitrary division of the 3D space that occurs when using a voxel grid. All of the candidate surface points are obtained from triangulating image samples with the available geometric constraints from camera calibration. Furthermore, the search space is restricted, because it is bounded by the visual hull as a result of the inclusion of silhouette information. As in space carving, every time some point has been carved away, the algorithm iterates on the new set of surface points. This is repeated until no more changes are detected, which means that the set of surface samples is globally photo-consistent.

After running this algorithm, we obtain the surface corresponding to a photo hull, represented by surface samples which are tighter to the actual surface than the initial candidates and can show concavities. In order to obtain a complete description of the surface, the orientation of these points must be estimated.

4.4 Orientation Estimation

In order to estimate the orientation of each surface point, a least squares method is applied for fitting a plane to a set of points in a neighborhood of it, assuming local flatness.

It is possible to fit a plane to a set of 3D points using least squares with errors measured orthogonally to the proposed plane. Let the plane be $\hat{\mathbf{n}} \cdot (\mathbf{x} - \mathbf{x}_a) = 0$, where $\hat{\mathbf{n}}$ is a unit length normal to the plane and $\mathbf{x}_a$ is a point on the plane. Define $\mathbf{x}_j$ to be the sample points; then

$$\mathbf{x}_j = \mathbf{x}_a + \lambda_j \hat{\mathbf{n}} + p_j \hat{\mathbf{n}}_j^* \tag{4.3}$$

where $\lambda_j = \hat{\mathbf{n}} \cdot (\mathbf{x}_j - \mathbf{x}_a)$ and $\hat{\mathbf{n}}_j^*$ is some unit length vector perpendicular to $\hat{\mathbf{n}}$ with appropriate coefficient p_j. Define $\mathbf{y}_j = \mathbf{x}_j - \mathbf{x}_a$. The vector from $\mathbf{x}_j$ to its projection onto the plane is $\lambda_j \hat{\mathbf{n}}$. The squared length of this vector is $\lambda_j^2 = (\hat{\mathbf{n}} \cdot \mathbf{y}_j)^2$. The energy function for the least squares minimization is $\mathrm{E}(\mathbf{x}_a, \hat{\mathbf{n}}) = \sum_{j=1}^m \lambda_j^2$. Two alternate forms for this function are

$$\mathrm{E}(\mathbf{x}_a, \hat{\mathbf{n}}) = \sum_{j=1}^{m} \mathbf{y}_j^\top \left(\hat{\mathbf{n}}\hat{\mathbf{n}}^\top\right) \mathbf{y}_j \tag{4.4}$$

and

$$\mathrm{E}(\mathbf{x}_a, \hat{\mathbf{n}}) = \hat{\mathbf{n}}^\top \left(\sum_{j=1}^{m} \mathbf{y}_j \mathbf{y}_j^\top \right) \hat{\mathbf{n}} = \hat{\mathbf{n}}^\top \mathbf{M}(\mathbf{x}_a)\, \hat{\mathbf{n}} \tag{4.5}$$

Using the first form of E in the previous equation, take the derivative with respect to $\mathbf{x}_a$ to get

$$\frac{\partial \mathrm{E}}{\partial \mathbf{x}_a} = -2 \left(\hat{\mathbf{n}}\hat{\mathbf{n}}^\top\right) \sum_{j=1}^{m} \mathbf{y}_j \tag{4.6}$$

This partial derivative is zero whenever $\sum_{j=1}^m \mathbf{y}_j = 0$ in which case $\mathbf{x}_a = \frac{1}{m} \sum_{j=1}^m \mathbf{x}_j$ (the average of the sample points).

Given $\mathbf{x}_a$, the matrix $\mathbf{M}(\mathbf{x}_a)$ is determined in the second form of the energy function. The quantity $\hat{\mathbf{n}}^\top \mathbf{M}(\mathbf{x}_a)\, \hat{\mathbf{n}}$ is a quadratic form whose minimum is the smallest eigenvalue of $\mathbf{M}(\mathbf{x}_a)$. This can be found by standard eigensystem solvers. The corresponding unit length eigenvector $\hat{\mathbf{n}}$ is the result of the extraction of the normal direction to a set of points by least squares fitting of a plane.

If $\mathbf{x}_a = (a, b, c)^\top$ and $\mathbf{x}_j = (x_j, y_j, z_j)^\top$, then matrix $\mathbf{M}(\mathbf{x}_a)$ is given by

$$\begin{pmatrix} \sum_{j=1}^m (x_j - a)^2 & \sum_{j=1}^m (x_j - a)(y_j - b) & \sum_{j=1}^m (x_j - a)(z_j - c) \\ \sum_{j=1}^m (x_j - a)(y_j - b) & \sum_{j=1}^m (y_j - b)^2 & \sum_{j=1}^m (y_j - b)(z_j - c) \\ \sum_{j=1}^m (x_j - a)(z_j - c) & \sum_{j=1}^m (y_j - b)(z_j - c) & \sum_{j=1}^m (z_j - c)^2 \end{pmatrix}. \tag{4.7}$$

The normal estimated by using this approach still shows one degree of freedom. The sign of the unit length eigenvector –the normal, $\hat{\mathbf{n}}$– corresponding to the smallest eigenvalue is not defined. Therefore, additional features have to be exploited to completely determine surface orientation. In the following, a mechanism for this image-based sampling strategy is presented.

4.4.1 Orientation in image-based sampling

Let r_n be an arbitrary maximum distance for neighbor points. The algorithm applied to each surface point $\mathbf{x}_i$ works as follows:

1. Define an extrinsic neighborhood ν_i including all the surface points at an Euclidean distance from $\mathbf{x}_i$ smaller or equal to r_n.

2. Estimate the mean position $\boldsymbol{\mu}_i$ of the neighborhood ν_i and modify the position of each neighbor point $\mathbf{x}_j$ by subtracting it:

$$\mathbf{x}_j := \mathbf{x}_j - \boldsymbol{\mu}_i, \quad \forall \mathbf{x}_j \in \nu_i$$

3. Create a matrix $\mathbf{M}_i = \sum_{\mathbf{x}_j \in \nu_i} \mathbf{x}_j \mathbf{x}_j^\top$. A vector $\hat{\mathbf{n}}_i$, orthogonal to the plane fitted by least squares to the neighborhood ν_i, is obtained as the first eigenvector of $\mathbf{M}_i$.

This method does not control whether the resulting orthogonal vector $\hat{\mathbf{n}}_i$ is inwards or outwards pointing, but the knowledge of the camera viewing the surface point helps setting the sign of the orientation by enforcing a negative dot product between the point normal and the director vector of the corresponding back-projected ray, which has a direction inverse to that from which the surface is visible.

4.5 Conclusions

The presented image-based approach to surface sampling obtains a set of colored and oriented surface samples, which describe the visible photo-consistent surfaces of the objects of interest in images –foreground objects– using an arbitrarily sparse set of views. With this technique, surface samples are found by a searching mechanism over regions defined as back-projected rays of pixels in every image.

As introduced in Chapter 3, it would be wise to exploit the information about the contents in a certain time instant to improve the quality or the efficiency of the reconstruction at the next time instant, due to the strong temporal correlation on the position of the surface samples, when working with multi-view video sequences. However, due to the selection of search regions for surface samples, an arbitrary motion of the surfaces in 3D space cannot be naturally modeled and incorporated in the proposed technique in order to enhance the efficiency or the accuracy of surface sampling.

As a summary, the presented technique makes a search of visible surface samples along the back-projected rays of each pixel in each view, in an image-based procedure exploiting epipolar constraints and photo-consistency between pairs of

neighbor views in arbitrarily wide baselines. Some valuable properties of this approach are:

- The visibility computation is implicit. Whenever a surface candidate is carved away, its next candidate becomes visible. This feature shows the method is suitable for the efficient computation of photo-consistent surfaces.
- It can provide a pixel-perfect projection of the reconstructed surface under accurate photometric and geometric camera calibration. This corresponds to the goal of being able to retrieve the original images by the projection of the reconstructed surfaces.

However, this approach presents some drawbacks that limit its efficient applicability for real-time processing of multi-view video sequences, which is the main target of this thesis:

- The reconstruction mechanism makes it difficult to exploit temporal correlations when reconstructing surfaces in multi-view video sequences, due to the strict definition of search regions along back-projected rays of pixels. This results in a limited throughput.
- Reconstructed surfaces are open, because only surfaces samples visible from at least one view are part of the final surface –surfaces are not extrapolated–. Therefore, algorithms exploiting topological properties of the objects surfaces will be affected by this limitation.

The next chapter proposes an alternative surface sampling approach that tackles these drawbacks by searching the positions of surface samples in every time instant around the deformation of an existing, closed surface from a previous time instant.

Chapter 5

Surface Sampling by Deformation

The previous chapter has introduced an image-based sampling scheme to obtain photo-consistent samples of visible surfaces in arbitrarily wide baseline setups. Such technique is able to extract a cloud of oriented points that can also describe the silhouette-consistent surface of a static scene, when replacing the color information by binary silhouettes. However, the design of the search strategy for surface samples is not suitable for exploiting temporal correlations in multi-view video sequences.

In order to cope with this limitation, a new sampling scheme based on the deformation of an existing surface is proposed, which naturally exploits temporal redundancies in addition to spatial ones.

5.1 Motivation

When multi-view video sequences are available, a large amount of information about the scene at each new time instant is known from the reconstruction at the previous time instant. Therefore, it seems straightforward to provide algorithms able to tackle the problem of 3D reconstruction as the exploitation of both spatial and temporal redundancies in multi-view video sequences simultaneously.

In contrast with the approach presented in the previous chapter, where the search regions for surface samples were back-projected rays of pixels, this time a different search strategy is considered, which is more oriented towards the actual structure of the surfaces to be reconstructed. The new search regions are defined

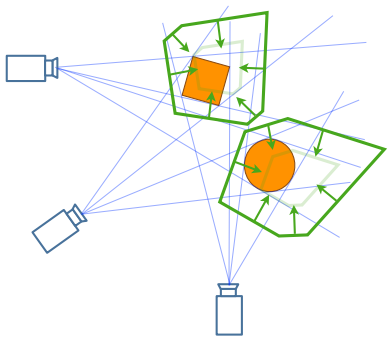


Figure 5.1: The volume enclosed by the dilation (dark thick line) of the existing surfaces (light thick line) of objects in certain time instant very likely contains the surfaces of these moving objects (in orange) in the next time instant, thus reducing the search space

along the (inward) normals of samples of an existing surface. This arbitrary choice exploits the fact that the volume enclosed by a sufficient dilation of an existing surface in a time instant most likely contains the object in the next time instant, under the assumption of a limited amount of motion between frames. This is illustrated in Fig. 5.1. The main advantage of this approach is that the search space for every time instant is greatly reduced, once conditioned to the knowledge of the previous position of surface samples.

5.1.1 Related work

A number of methods exists for extracting the *visual hull* (VH) from a set of binary silhouettes. Some provide a voxelized volumetric representation [Cheung et al., 2000] and others provide surfaces described by polygonal meshes [Franco and Boyer, 2009]. The VH is fast to estimate and offers an initial description of the volumes in the scene for applications with camera rigs in arbitrarily wide-baseline setups. However, these algorithms are not designed to exploit temporal redundancies available in video sequences.

Some achievements in motion estimation have been made by separating the shape and motion estimation tasks, in a *scene flow* estimation approach of the problem that can use color [Vedula et al., 2005] or spherical models [Starck and Hilton, 2005] as matching features, although the shape estimation problem is treated statically, without exploiting temporal redundancies. There also exists a growing number of algorithms that jointly estimate the 3D motion and photo-consistent shape of scenes by tracking surfaces in multi-view settings [Furukawa and Ponce, 2008, Courchay et al., 2009, Goldluecke and Magnor, 2004] using local normalized cross-correlation and smoothing constraints, a global variational approach or spatio-temporal hypersurfaces, respectively. As a result, impressive time-coherent

reconstructions are obtained at the cost of large computation times. [Guan et al., 2008] tracks people in crowded scenes, exploiting image-based appearance models, which help disambiguating shape modeling without requiring color calibration between cameras. This approach is designed to work even in wide-baseline setups.

Using range data for initializing a high-quality mesh model of the surface of a person, [de Aguiar et al., 2007] applies Laplacian deformations in order to fit the high-quality surface to the set of silhouettes at each instant, obtaining impressive off-line results. However, in *smart room* environments it is desirable to only consider the data obtained from calibrated cameras, with known or static background, in order to make a better user experience. In such a scenario, no range data is available in order to estimate precise models of the dynamic objects in a scene.

5.1.2 Strategy

The methodology for efficiently extracting silhouette-consistent surface samples by the exploitation of temporal redundancies that is presented next corresponds to a tracking strategy as opposed to *scene flow* estimation strategies. The proposed strategy is divided in two parts:

- **Initialization: Static reconstruction**. First, samples of a silhouette-consistent surface are obtained for an initial time instant without having any knowledge of the shape at previous time instants by exploiting only spatial redundancies.
- **Tracking: Dynamic reconstruction**. Then, the knowledge of the shape in the previous time instant allows for a more efficient sampling of the surfaces in the next time instant, by exploiting both spatial and temporal redundancies.

The different stages composing this technique for surface sampling, based on the deformation of an existing surface, are illustrated in Fig. 5.2. In both stages of the complete tracking strategy –initialization and dynamic surface–, the resulting output is a sampled representation of the surfaces of the VH at every time instant.

5.2 Initialization: Static Reconstruction

The starting point of the initialization procedure is the definition of an arbitrary, closed initial surface that encloses a large volume containing the scene to be reconstructed. The initial surface is described as a point cloud of arbitrary density with the property that each point p_i has a unitary displacement vector $\mathbf{m}_i$, orthogonal to the initial surface, pointing inwards to another point p_i^*, also lying on the surface.

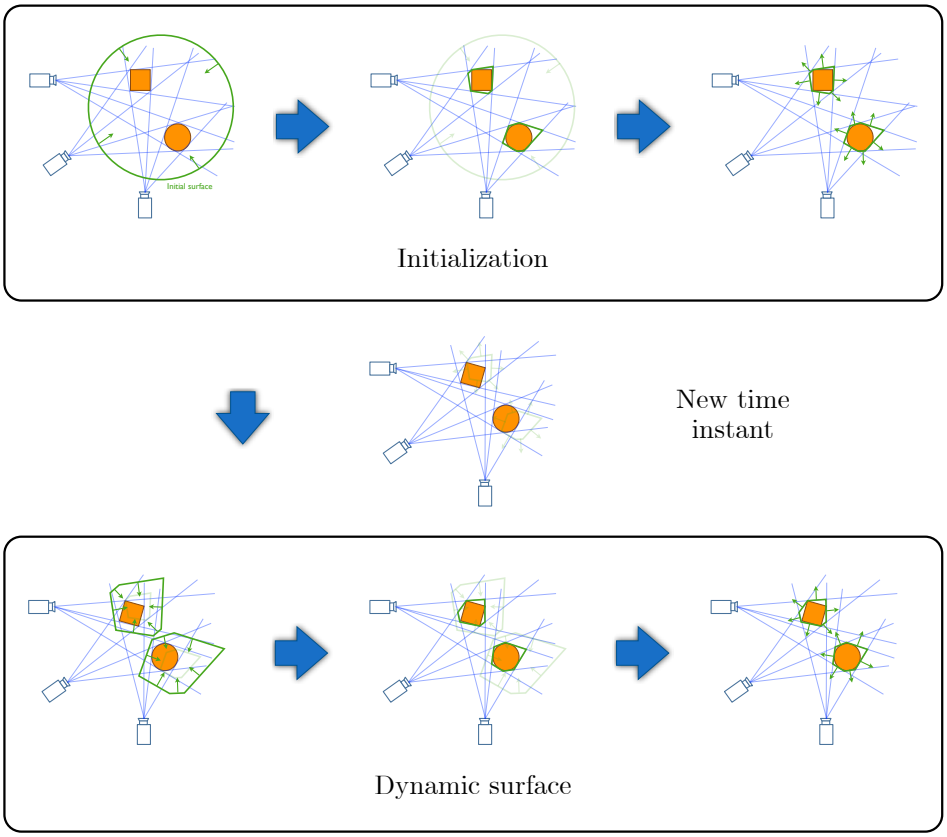


Figure 5.2: The tracking strategy is divided in an initialization stage, at which an arbitrary sampled surface is deformed to represent the surface of the foreground objects of the scene, and a tracking stage (dynamic surface), at which the surface in the previous time instant is used in order to estimate more efficiently the position of surface samples in the new time instant

This property defines a segment s_i for each pair of surface points p_i and p_i^* in which their displacement range is constrained.

This arbitrarily defined constraint on the displacement range of surface points is suitable for two reasons. On the one hand, it ensures that the processing of a new view keeps each surface point inside of the limits imposed by any other previously processed viewpoint. On the other, it allows to keep topological information about interior and exterior of the surface.

The main concept behind the proposed algorithm for the initialization stage is the iterative introduction of modifications of the closed surface in order to fit its projection to the foreground silhouette of each viewpoint in the multi-camera setting. This concept in spatial domain resembles that of adapting an existing surface to a new set of silhouette constraints in another instant in time domain, which is the principle of the tracking part of the presented surface sampling approach.

At the output of the static reconstruction stage, oriented silhouette-consistent

surface samples, corresponding to a Visual Hull, are obtained. First, silhouette-consistency is imposed by a per-view application of a set of topological operations on each segment connecting surface points and a global merging of the resulting segments. Then, orientation is estimated by exploiting local distributions of surface samples and knowledge about the interior and exterior of the objects, given by the segments.

5.2.1 Silhouette constraints

Silhouette constraints are imposed in a two-staged approach that copes with the problem of camera settings not offering complete views of the scene to be reconstructed. First, in a per-view stage, every segment is conceptually shrunk and cut into smaller pieces by imposing the silhouette constraints corresponding to each view. Then, a global combination of these partial results delivers the final set of surface points, which are the extremes of such segments.

Per-view constraints

The first stage when adapting the arbitrary initial surface –described as a point cloud with connected points forming segments– to a new silhouette is the *shrinkage* of each segment s_i. Both extrema p_i and p_i^* are displaced along the segment towards each other until they either intersect the back-projected silhouette or meet each other. In the former case, the segment is just shrunk in order to fit the silhouette constraint. In the latter case, the segment –and therefore its extreme points– can be discarded.

However, after shrinking a segment, holes in silhouettes and empty regions between silhouettes still have to be accounted for. In this case, the extremes are further moved towards each other until they meet, introducing additional segments by *cutting* the current segment every time the point displacement along the segment makes them enter and leave the interior of the back-projection of the silhouette. These two topological operations on the segments are illustrated in Fig. 5.3.

After these two operations, each segment is replaced by a set of new, shorter segments that fit the constraints of the silhouette corresponding to the view under consideration. When daisy-chaining this stage between all views, these new constraints are automatically merged to those resulting from the silhouettes of previously processed views. Throughout the entire process the surface remains closed.

However, as seen in the next section, it is better to consider this daisy-chained version of the method only as a concept, for it would be faster to execute –each view reduces the search space of the subsequent views in the chain– but would not

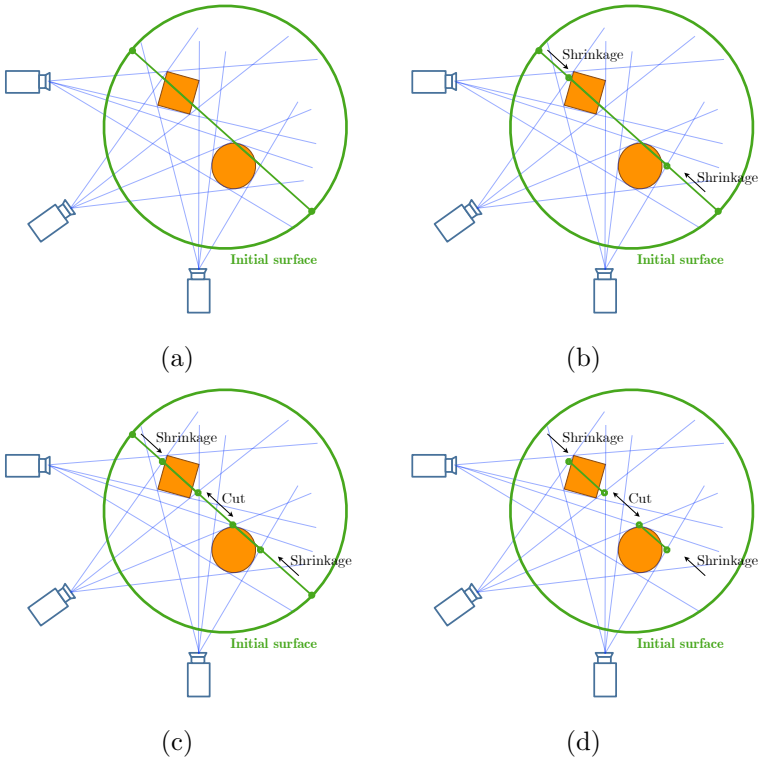


Figure 5.3: Topological operations on a segment connecting two surface points by the imposition of the silhouette constraints from one view. (a) Initial segment connecting two points on an arbitrary initial surface. (b) Shrinkage in order to fit the visible extents corresponding to the silhouettes in that view. (c) Segment is cut in two pieces due to the silhouette configuration. (d) Resulting set of segments for this view

provide mechanisms to reconstruct parts of the foreground objects that lie out of the *frustum* –or viewing cone– of one or more views.

Global combination

Rather than directly performing the topological operations –in other words, cutting and shrinking segments– during the processing of each input silhouette, an ordered list of events for each initial segment accumulates both the partial inclusion in the camera's frustum and the *occupancy* from each view. The types of events are illustrated in Fig. 5.4. A final stage merges the per-view constraints and decides the topological operations applied to each segment. The resulting algorithm processes each initial segment as follows:

1. For each view, the events corresponding to the partial frustum inclusion and occupancy extremes –shown in Fig. 5.5– are obtained.

2. The events from all views are stored in a list, ordered by the distance of the

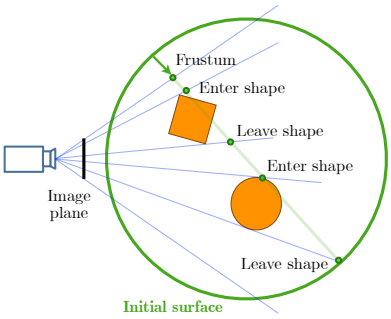


Figure 5.4: 2D representation of a set of events detected along a segment from a certain viewpoint

position of each event to a pivot point on the line where the segment lies.

3. The extremes of the resulting segments are those points where either the occupancy count becomes equal to the number of inclusions in frusta (entering shape) or the occupancy becomes smaller than the number of inclusions in frusta (exiting shape), as shown in Fig. 5.6.

The resulting surface points are the extremes of the segments created during the global combination stage. Using this mechanism, the topological operations on the segments joining points from the initial closed surface are applied at the last stage. This way, it is not necessary that the object is included in the frustum of each view in order to be reconstructed.

Furthermore, *conservative* estimates of the surface of the visual hull are possible, by assuming the existence of τ silhouette under-segmentation errors of a point included in the volume of a foreground object –in other words, in τ views, the projection of the same part of an object is classified as background–. Then, in the third step of the previous algorithm, a tolerance τ can be used as a sufficiently small difference between the occupancy count and the number of inclusions in frusta that produces a surface point. $\tau = 0$ provides the non-conservative, visual hull surface.

In order to characterize the surface by a set of points, without keeping topological information included in the extents of the segments, the orientation of each surface sample must be defined as shown next.

5.2.2 Orientation estimation

Normals are geometrically estimated in a way similar to that in the image-based method in Section 4.4. However, in order to estimate whether the resulting vector $\hat{\mathbf{n}}_i$ is inwards or outwards pointing, the knowledge of the unitary displacement vector of each surface point –pointing towards where the segment lies– is the tool

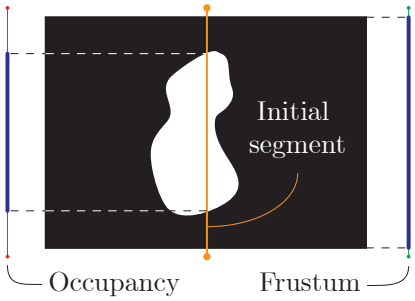


Figure 5.5: Partial frustum intersection and occupancy of an initial segment from a viewpoint. The orange segment represents the projection of a 3D segment onto the image plane.

used to determine the sign of the estimated normal. Using the same notation as in the image-based case, let λ_i be the average projection norm of the inward-pointing unitary director vector $\mathbf{m}_j$ tied to each $\mathbf{x}_j \in \nu_i$ onto $\hat{\mathbf{n}}_i$. If $\lambda_i < 0$, $\hat{\mathbf{n}}_i$ is considered to point outwards and therefore its sign remains unchanged, whereas, if $\lambda_i > 0$, it is forced to point outwards by changing its sign: $\hat{\mathbf{n}}_i := -\hat{\mathbf{n}}_i$. In Fig. 5.7, a set of surface points lit by a a virtual light at the viewing position is shown. At the left side normals are not available and, therefore, all points are lit equally, whereas at the right side the use of normal information provides a better representation of the surface.

The neighborhood ν_i used for estimating the normal direction also provides an approximate description of the local topology of the reconstructed surface around each sample. This information proves useful for the dynamic estimation of the position of surface samples, which will require the application of spatial regularization.

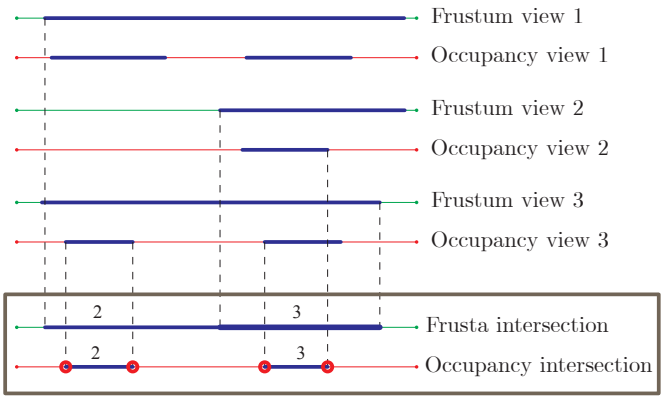


Figure 5.6: Occupancy along an initial segment for 3 input views with inclusion in a minimum of 2 frusta. The resulting surface points are the extremes of the occupied segments

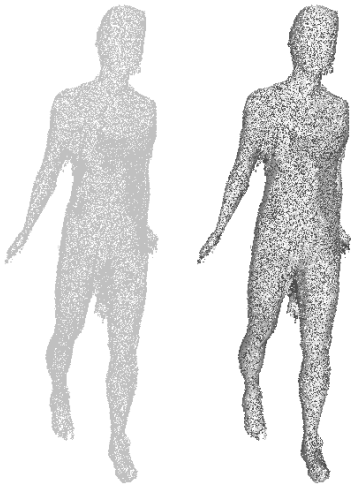


Figure 5.7: Surface points lit considering normals (right), which are geometrically obtained from the positions of sample points on the surface of the visual hull (left)

5.3 Tracking: Dynamic Reconstruction

As introduced in the previous chapter, the image-based reconstruction technique defines an arbitrary search region for each surface element, which is the back-projected ray corresponding to each pixel of the objects of interest –foreground objects–. This search criterion keeps the method from naturally handling shape priors in sequences with motion, due to the fact that search areas cannot be modified in order to exploit directions of motion of a surface sample between two consecutive time instants different from the arbitrarily defined ones.

However, a surface deformation-based method allows for the choice of a better adapted search region for each surface sample. Hence, the design for the search region is exploited in this section in order to accept shape priors from a previous time instant and track them at each new time instant. The proposed method is the first one in this thesis exploiting spatial correlation. It is capable of obtaining a surface sampling at each time instant equivalent to that obtained with static approaches (either the initialization stage of this technique or the image-based approach in the previous chapter), but with the benefit of a smaller computational cost.

In the initialization stage of this technique, parts of the initial arbitrary surface –represented as a point cloud with pairs of points defining segments– were allowed to vanish in cases where the strictly guided displacement of the surface points along their arbitrarily defined search regions makes the extremes of a segment collapse without having crossed a surface along their path. However, when attempting to estimate shape in a tracking strategy, it is not permissible to discard surface points, since that would result in a growing under-sampling of the surfaces as time passes. Therefore, the proposed method introduces a regularization strategy for reducing

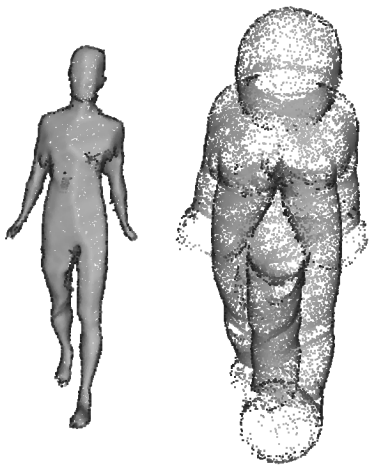


Figure 5.8: A sampled surface and its dilation following the estimated outward normals for each surface sample. The size of the surface samples has been enlarged for clarity

the side-effects of arbitrarily defining the search regions for each surface sample.

5.3.1 Surface expansion

Assuming that the displacement of surface samples between consecutive frames is bounded, r_t is defined as an arbitrary radius of the 3D search area from one time instant to the next. The search space is defined in spatial units of the 3D reference system, instead of pixel units that would be used in classical image processing. This is a clear advantage of working with 3D reconstructions of multi-view data, which is fundamental in the proposal of this thesis.

The dilation of a sampled surface can be defined as the displacement of each surface sample by a parametric distance r_t in the direction of the normal of each surface sample. Fig. 5.8 depicts the *dilation* of a surface obtained by translating the surface points a distance r_t following their estimated outward normals. It is interesting to note that the volume enclosed by the resulting surface very likely contains the object at the next time instant. Thus, r_t controls the range of distances a surface sample can displace between from frame to frame such that it can be tracked by the proposed algorithm. However, a mechanism for recovering samples displaced a distance larger than r_t is also considered by means of spatial regularization.

5.3.2 Spatial regularization

Similarly to the initialization procedure resulting in a static reconstruction, the search direction for each surface sample is again set as an inner pointing vector –in this case, the inverted normal of a surface sample–. The main difference is that in

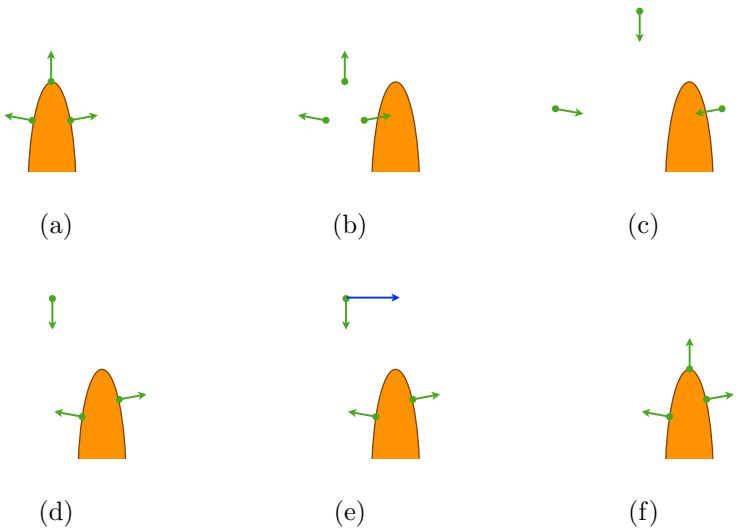


Figure 5.9: In (a) a set of oriented surface samples (green) represents the surface of the orange shape. In (b) the shape has moved with respect to its initial position. In (c) the initial surface samples have been dilated along their normals in order to generate a new search volume. In (d) some samples reach the surface following the inward normal direction. In (e) local rigidity is imposed in the remaining sample by adding a tangential displacement (blue arrow). In (f) the whole set of samples lies on the new surface thanks to the use of regularization

this case the search space is greatly reduced with respect to that of the initialization. Instead of an arbitrary surface enclosing the volume where foreground objects lie, the enclosing surface is the result of dilating the surface in the previous time instant.

As shown in Fig. 5.9, the arbitrarily defined search direction results in some surface points not being able to meet the surface in the previous time instants, whereas others slide over the surface. In order to cope with the former problem and reduce the latter, regularization is applied in order to enforce local rigidity, as seen below.

Let $\mathbf{x}_i^e$ be a surface point expanded along its normal $\hat{\mathbf{n}}_i$ by r_t units. Its inward normal direction $-\hat{\mathbf{n}}_i$ can be considered as a direction to look for an initial estimate of its position at the new time instant. However, some sort of regularization is needed in order to keep local rigidity on the set of points. Therefore, an iterative method with motion feed-back is applied on the whole list of points. In outline, this iterative method modifies the search range of each point $\mathbf{x}_i^{t-1}$ by adding a set of tangential displacements d_i obtained from its neighbors ν_i. In more detail, the initial set of displacements for each surface point is initialized $\mathbf{d}_j := (0,0,0)^\top, \quad \forall \mathbf{d}_j \in d_i$. Then, the following method is applied to each surface point $\mathbf{x}_i$ iteratively:

1. Let N_i be the number of $\mathbf{x}_i$'s assigned displacements. Define an initial *displaced* position as the addition of its position at the previous time instant and its current set of displacements $\mathbf{x}_i^d := \mathbf{x}_i^{t-1} + \frac{1}{N_i} \sum_{\mathbf{d}_j \in d_i} \mathbf{d}_j$.

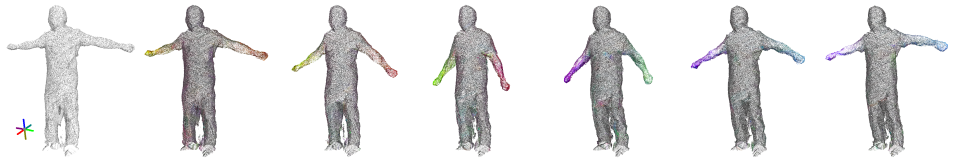


Figure 5.10: Shape and motion jointly estimated from sequences of silhouettes extracted from 8 cameras placed around a person. The estimated velocity is color encoded following the axes at the bottom-left side. White points correspond to the initial surface.

2. Define a virtual segment, joining its displaced expanded position $\mathbf{x}_i^e := \mathbf{x}_i^d + r_t\hat{\mathbf{n}}_i$ and its displaced *contracted* position $\mathbf{x}_i^c := \mathbf{x}_i^d - r_t\hat{\mathbf{n}}_i$.

3. Shrink the virtual segment by displacing the *outer* surface point $\mathbf{x}_i^e$ towards $\mathbf{x}_i^c$. Along this path, store the closest position to $\mathbf{x}_i^{t-1}$, namely $\mathbf{x}_i^s$, at which the outer surface point crosses the surface and enters the intersection of the back-projection of all the available silhouettes corresponding to cameras with frusta containing that region.

4. If the $\mathbf{x}_i^s$ is found, keep it as the current candidate for its new position $\mathbf{x}_i^t := \mathbf{x}_i^s$ and obtain the corresponding displacement $\boldsymbol{\delta}_i := \mathbf{x}_i^t - \mathbf{x}_i^{t-1}$.

5. If $\boldsymbol{\delta}_i$ is defined, set a displacement term for each neighbor $\mathbf{x}_j \in \nu_i$ orthogonal to its displacement direction $\boldsymbol{\tau}_{ij} = \boldsymbol{\delta}_i - \boldsymbol{\delta}_i \cdot \hat{\mathbf{n}}_j$. The motion component along the inward normal is directly set by the shrinking mechanism.

The regularization procedure described above is executed until no more samples can be fit to the new silhouettes. Some samples, lying in regions such as limb extremes, might not be attached to the surface in the new time instant, despite of the regularization. However, they can be integrated in the reconstructed surface by applying the same rigid transform (in this case a translation) as the resulting one of its closest neighbor that fits the new surface. This method allows these surface samples to be recovered at subsequent iterations of the algorithm in another time instant, without the need to destroy them and create new ones in order to prevent an otherwise progressively growing uncovered area of the surface.

Finally, the normal estimation method introduced in static reconstruction (Section 5.2.2) is applied to each of the resulting surface points. A dense velocity field can be obtained as a first order estimate by simply taking the displacement of each surface point between two consecutive time instants. This velocity field is not intended to be accurate, but it illustrates the direction and magnitude of the displacements that result after the application of the dynamic method. In Fig. 5.10, it can be observed how the color-encoded velocity field estimated in this manner actually provides a plausible approximation to the actual motion of the scene.

5.4 Photo-consistency Constraints

So far we have seen how silhouette constraints can be used in order to deform an initial surface and make it represent the surface enclosing the visual hull of the set of silhouettes. However, color information contained in the images captured by the multi-camera rig has not yet been introduced to the surface deformation mechanism.

The approach we propose in order to introduce the photo-consistency constraints imposed by the contents of the color images is that of further deforming the resulting surfaces until they are photo-consistent with the complete set of available views. The approach is divided in two steps that are repeated iteratively until no more segments are shrunk in order to fulfill the photo-consistent constraints imposed by the available images:

1. Obtain the visibility of each surface point.

2. Shrink each segment until its extremes (the surface points) are either non-visible or photo-consistent.

5.4.1 Visibility estimation

Computing the visibility of a surface point from a given viewpoint is not straightforward, due to the fact that the topology of the surface is unknown. Furthermore, the density of the point cloud describing the surface can be arbitrarily small. Even if the exact topology of the surfaces in the scene remains unknown, the neighborhood of each surface point provides useful information for obtaining its visibility.

The condition for a surface point to be visible from a certain viewpoint is that it has a direct line of sight with the center of projections of the camera. Let $z\,(p = (x, y))$ be the depth map of a certain viewpoint. If a surface point is visible, its depth measured from the center of projections must be equal to the depth of the pixel onto which it projects for this cam.

The problem that arises is that of computing a proper depth map from a cloud of 3D points projected onto a given viewpoint without an exact knowledge of its topology. In practice, this means that an area around the projection of each surface point has to be drawn in order to close the depth map and avoid the possibility that a surface point in a hidden surface becomes visible through the holes. The chosen approach consists in drawing the convex hull of the neighborhood of each surface point, which ensures filling in the gaps between projected surface points. The resulting algorithm for each view works as follows. First, initialize the depth map to a value further than any surface point. Then, for each surface point:

1. Compute the depth of the surface point with respect to the viewpoint z_p.

2. Compute the convex hull of its neighborhood.

3. For each pixel in the convex hull, if its depth is smaller than the value stored in the depth map, set its depth to z_p.

Once the depth map is computed, each surface point whose depth with respect to a viewpoint is smaller than $z(p) + \delta$, with $p = (x_0, y_0)$ and δ a threshold for balancing the inaccuracy of the convex hull representation empirically set to $1/2r_n$, is considered to be visible from that viewpoint.

The depth map can be improved by computing the exact topology of the surface and drawing faces instead of convex hulls. Doing so, the δ threshold would be unnecessary and the visibility computation would be, as a result, more precise.

5.4.2 Segment shrinking

When an extreme of a segment is visible from more than two views, its photo-consistency has to be measured, using the same method as in the image-based method (Section 4.2). When the photo-consistency test is not passed, the segment starts a shrinking process until it either becomes hidden, turns photo-consistent or vanishes by meeting its other extreme.

When all visible segments have been processed in this manner, a new carving iteration starts. First, the current visibility of each view is computed and, then, the segments are shrunk accordingly to their visibility and photo-consistency. The algorithm ends when all the extrema of the segments, *i.e.* the surface points, are photo-consistent.

5.5 Conclusions

A new surface sampling strategy has been presented, which exploits temporal and spatial correlations in order to reduce the computational cost associated to the reconstruction of each frame in scenarios with static, known background. This technique is based on the deformation of a closed initial surface in order to obtain silhouette-consistency.

The design of the technique has been driven by the proposal of solutions to the problems found in the sampling strategy presented in the previous chapter. However, the algorithm presented in this chapter shows some drawbacks:

- Obtaining pixel-perfect projections requires a high number of initial surface points due to the random initial placement of surface points, many of which are discarded as a result of the deformation strategy for initialization.
- Photo-consistency constraints require costly visibility computations when compared to the technique presented in the previous chapter.
- The initialization part of the technique is not well-suited to massively parallel platforms, such as GPUs, due allocation and memory alignment issues derived from the division of the initial segments connecting two surface points into an undetermined quantity of smaller segments resulting from the two applied topological operations.
- There exists a long-term drift of surface samples towards areas further apart from the motion direction, due to the action of rigidity constraints –which are, in turn, necessary in order to recover surface samples close to extremes, such as limbs, and avoid excessive sliding of points on the surface–.

On the other hand, this method shows important advantages with respect to the technique presented in the previous chapter:

- The reconstruction mechanism allows exploiting spatial and temporal correlations on the position of surface samples in multi-view video sequences, thus improving the efficiency of the sampling stage.
- The search space is not determined by the images, but rather by the 3D structure of the scene.
- The reconstructed surfaces are closed, with extrapolated non-visible parts, which provides a more effective representation for visualization –*stencil buffer shadowing* [Segal et al., 1992]– and analysis applications.
- The strategy might be used to obtain an initial estimate of the dense motion of the scene. Although this is not the main target of the sampling approach, it could be used as an additional feature in multi-modal analysis applications for motion detection.

Chapter 6

Statistical Surface Sampling

The previous chapter has presented a sampling approach intended for the exploitation of temporal correlations in multi-view video sequences. Although spatial correlations are implicitly exploited in the regularization process taking place in the tracking stage of that algorithm, in this chapter we present a methodology for taking advantage of spatial correlations in a more principled manner.

Samples of 2D surfaces in 3D space show a high degree of spatial compactness. Indeed, given a certain volume in 3D space which dimensions are in the order of $\propto r^3$, its corresponding 2D enclosing surface has dimension $\propto r^2$. This result implies that, given a known surface sample, new surface samples can be efficiently found by inspecting a compact region surrounding it.

In this chapter, this property is exploited in order to improve the process of surface sampling from a set of calibrated views. First, we motivate the exploitation of spatial correlation. Then, we present an initialization or *scouting* stage, in which a small number of *seed* surface samples is obtained, and a propagation stage, where a dense set of surface samples is efficiently obtained by exploiting spatial correlation. In order to improve the efficiency of the initialization stage, two techniques are proposed, which exploit temporal correlations in multi-view video sequences and multi-resolution. Finally, some techniques for the post-processing of the resulting set of surface samples are presented.

6.1 Motivation

The working principle of the technique presented here can be justified by an example with lower dimensionality. Let p_j denote a pixel and $i(p_j)$ a binary 2D image with

$i(p_j) = 0$ for pixels representing an empty area and $i(p_j) = 1$ for pixels representing an occupied area, with an image size equal to $a_t = \sigma_x \times \sigma_y \propto r^2$.

Let $a_b \propto r$ be the size of the border of such region. Then, if we define a procedure where p_j takes a random pixel in the rectangle $[1..\sigma_x] \times [1..\sigma_y]$, the probability of one such pixel lying on the border of the occupied region, π_b, is the ratio between the size of the border and the size of the complete image, or $\pi_b = a_b/a_t \propto r/r^2 \propto 1/r$. This procedure is illustrated in Fig. 6.1 (a).

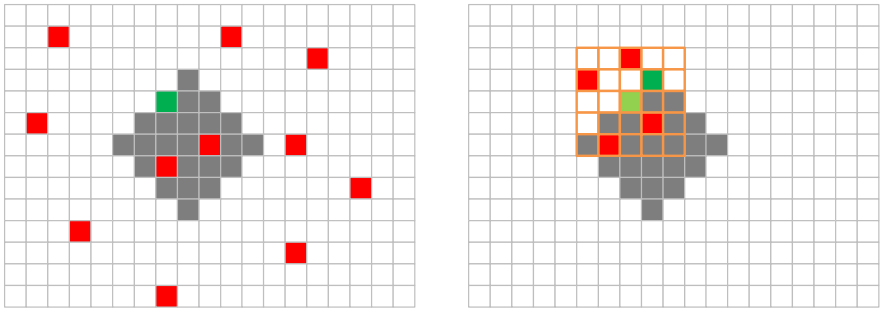
However, after a large number of attempts $O(1/\pi_b)$, the probability that the result of at least one of these pixels lies on the border of the occupied region tends to one. Although the number of attempts could be prohibitively expensive if the procedure was to be applied for each border point, it could be affordable at an initialization step if then a better suited strategy follows.

Let δ_n be a representative distance between neighbor points, and $\rho_n \propto \delta_n$ a search radius. A second procedure is defined as follows: let p_b be a border pixel obtained after a large number of tries of the first procedure and generate a new p_j as the sum of p_b and a random value in the ball $p'_j : \|p'_j\| \leq \rho_n$. Since the border in a local region is approximately a straight line, the number of border points is $\propto \rho_n$, whereas the area of the search region is $\propto \rho_n^2$. This means that the probability that the picked up pixel lies on the border of the occupied region, π'_b, is the ratio between the size of the intersection of the border with the new search region and the size of this new search region, or $\pi'_b \propto \rho_n/\rho_n^2 \propto 1/\rho_n$. Since $\rho_n \ll r$, the resulting success probability for the new procedure accomplishes $\pi'_b \gg \pi_b$. This second procedure is illustrated in Fig. 6.1 (b).

With this example we have shown how the search in local regions can provide an efficiency improvement with respect to global searches. The same idea can be translated to a higher dimensional space, which results of great interest for the surface sampling problem. In the following, the details of the application of this principle in the extraction of surface samples is explained in detail: the strategy for obtaining the random surface sampling consists in an initial scouting procedure. As a result, a collection of *seed* surface points uniformly distributed over the surface is obtained. Finally, a finer local search provides a dense sampling of the object surfaces. The challenge of this method is to guarantee that surface parts are not missing due to the non-exhaustive search of surface points.

6.1.1 Random sampling in multi-view environments

Similarly to the case of finding a 1D contour in a 2D space, in multi-view environments it is necessary to find 2D contours (surfaces) in a 3D space, with the difference that, in this case, the only data from which we have actual measurements



(a) (b)

Figure 6.1: Random sampling of a 1D contour in a 2D space. (a) It takes a large number of random tries to find a contour sample. (b) Once a contour sample has been found, a proper neighborhood (pixels delimited by orange lines) around a given contour point (light green pixel) helps in reducing the number of tries to find another contour sample.

are the 2D projections of the 3D scene onto each available view. Thus, 3D points under study have to be evaluated from their 2D projections onto each view.

6.1.2 Silhouette-consistent surface reconstruction

Given a set of silhouettes corresponding to different views of a scene, a silhouette-consistent surface point necessarily projects onto a silhouette contour pixel in at least one of the images. In order to validate this statement, please keep in mind that the occupied region of the visual hull [Laurentini, 1994] is uniquely defined by a set of silhouettes. A property supporting this statement is the fact that the back-projection of a silhouette point contacts the actual surface of the object in at least one point.

Therefore, the problem of reconstructing the silhouette-consistent surface by random sampling can be tackled in the following manner. Let $\mathbf{P}$ be the outcome of a random experiment that generates 3D points inside of a predefined bounding volume $v_t = \sigma_x \times \sigma_y \times \sigma_z$. Let now $\mathbf{p}_c$ be the 2D projection of $\mathbf{P}$ on a view c. If $\mathbf{p}_c$ belongs to the contour of the silhouette in a camera c, then $\mathbf{P}$ is part of the surface. Naturally, the efficiency of this experiment, which is the base for the *scouting* or *initialization* stage, is rather low.

However, the actual improvement introduced by random sampling is not shown by this experiment. The search efficiency can be improved by reducing the search range. If we define a proper neighborhood around a surface sample, a second experiment can generate random 3D points $\mathbf{P}'$ lying inside. This corresponds to the second stage, *propagation*, of the proposed method. In this stage, the likelihood that a randomly generated sample $\mathbf{P}'$ belongs to the surface will be much greater than in the initialization stage.

6.2 Initialization

The algorithm for silhouette-consistent reconstruction consists of two stages. The first stage, *scouting*, is responsible for generating a collection of *seed* surface samples by random sampling the space enclosed by a bounding box where foreground objects are to be found. The second stage extracts new surface samples by iteratively testing new points in suitable neighborhoods around already obtained surface samples.

The concept of propagation can be understood as that of assigning a property of a certain sample to others lying in its neighborhood. As a result, a sparse initial set of samples of a certain class can be expanded to a dense set of samples. In our method, points lying around seed surface samples can be selected as new surface samples after a local propagation, as introduced in the next section.

The decision of generating more than one seed surface sample is due to the chosen type of propagation scheme, which is introduced below. Propagation can be done in two different manners:

- *Omni-directional.* Each surface sample propagates with equal likelihood in every direction inside a suitable neighborhood.
- *Directional.* Each surface sample is given a propagation direction such that the propagation process does not result in new samples in regions that have already been visited.

In practice, since the location of the surfaces is unknown, the only possibility is to apply the former. In order to apply the omni-directional propagation scheme without introducing local oversampling, it is necessary to define a uniformly distributed collection of surface samples that will propagate to a dense set of surface samples after an adequate number of iterations. The details will be seen in the next section, but for now we keep in mind the idea of obtaining a sparse set of uniformly distributed seed surface samples.

Let $[m_x..M_x] \times [m_y..M_y] \times [m_z..M_z]$ be a bounding box enclosing the scene to be reconstructed. An initial random experiment is defined, which produces random 3D points inside of this bounding box. Then, in order to accept a generated 3D random point $\mathbf{P}$ as belonging to the surface of one of the objects to be reconstructed, three conditions have to be fulfilled:

1. The 3D point projects onto a foreground pixel for each available view.
2. The 3D point projects to a contour pixel for at least one of the views.
3. A normal (the point's orientation) can be estimated by operations on a close neighborhood of the 3D point.

When these three tests are passed by a random 3D point, this is accepted as a seed surface sample after attaching its orientation estimation.

6.2.1 Surface point validation

Let N_c be the number of cameras with *frustum* including the 3D point under evaluation or, in other words, the number of cameras for which the projection of the 3D point $\mathbf{P}$ lies inside of the corresponding image. If for one of these N_c cameras, the pixel where $\mathbf{P}$ projects to belongs to the background of the scene, then the point is discarded.

The projection test described above checks the first of the three conditions for accepting the 3D point as belonging to the surface. In order to ensure that $\mathbf{P}$ is an actual surface point, a second test is also performed on the point. Given a pixel $\mathbf{p}_c$ where the 3D point $\mathbf{P}$ projects onto the image corresponding to camera c, we test if at least one of the pixels in its 4-neighborhood belongs to the background. If the test is positive for at least one of the cameras, the point is accepted for testing the third acceptance condition.

Conservative estimation. This surface point validation method is correct in absence of segmentation errors. In case that there are some pixels wrongly marked as belonging to the foreground, the consistency checks between a large enough number of cameras will most likely reject false surface points projecting onto one of such pixels. The case of false background detection, however, will introduce severe reconstruction errors.

Assuming segmentation errors are uncorrelated between views, we can derive that the presence of one such an error in one of the N_c cameras onto which a 3D point projects is most surely accompanied by $N_c - 1$ correct decisions in the remaining cameras. Thus, by setting a tolerance to one segmentation error, a better estimate of the actual object can be obtained from noisy foreground detections, at the cost of slightly wider reconstructed shapes, providing a conservative estimate of the silhouette-consistent surface.

6.2.2 Local normal estimation

The orientation of each seed surface sample is important for two reasons. On the one hand, it offers necessary information about the topology of the surface which will be useful in order to interpolate the surface, as presented in Part II of this thesis. On the other hand, and more importantly for the approach presented in this chapter, it is necessary in order to define suitable search regions for the propagation

stage. In order to estimate the orientation of a surface point, a local cluster of surface points is estimated in a close neighborhood of a valid surface point. Given a valid surface point $\mathbf{P}$, a new random search is performed in a ball of radius ρ_n centered at $\mathbf{P}$. ρ_n is an adjustable parameter that defines a small distance such that the local surface remains practically flat (constant normal). In general, it is defined as a small fraction of the main diagonal of the bounding box of the working volume.

Three new points are generated in this neighborhood, which pass the surface point validation test described above. Then, a plane is fitted to the set of 4 points by *least squares* and the director vector of the plane is taken. As introduced in Section 4.4, doing so leaves us with a degree of freedom: the sign of the normal vector. In order to determine its sign, an iterative method is applied, which consists in the following steps:

1. Add the orientation of the surface point multiplied by a small scaling value to the position of the surface point $\mathbf{x}_+ := \mathbf{x} + \alpha\mathbf{n}$.
2. Subtract the orientation of the surface point multiplied by a small scaling value to the position from the surface point $\mathbf{x}_- := \mathbf{x} - \alpha\mathbf{n}$.
3. Obtain the results of the projection tests for both $\mathbf{x}_+$ and $\mathbf{x}_-$.
4. If one of the projection tests results in empty space and the other results in occupied space, the point can be accepted. If $\mathbf{x}_+$ belongs to empty space, the normal's sign is already correct. Otherwise, we change its sign: $\mathbf{n} := -\mathbf{n}$.
5. If both projection tests have the same result and $\alpha > \alpha_{\min}$, repeat the process with $\alpha := 4/5\alpha$. If $\alpha \leq \alpha_{\min}$, the point cannot be successfully oriented.

If at the output of this test the result is that the 3D point $\mathbf{P}$ could not be successfully oriented, it is discarded, together with its local neighbors.

6.3 Propagation

From the collection of seed surface points, which are sparse but uniformly distributed over the surface, a propagation procedure is defined in order to efficiently provide a dense sampling of the surface by exploiting spatial correlation. Thus, the following method aims at obtaining new surface samples at smaller distances from each other, resulting in a denser surface sampling. The benefit of introducing new samples taking into account the position of already obtained samples is the statistical reduction of the number of random tries until we find new valid surface samples.

6.3.1 Propagation region

Given a known surface sample, we define a local 3D region around it where new surface samples are very likely to be found, given that surfaces are continuous. The design rules of this propagation region are the following:

- Assuming surfaces are locally flat, or present a large curvature radius, preference will be given for new surface samples to be placed close to the plane defined by the surface sample's position and orientation.

- Given the fact that a certain density of surface samples is already placed around each surface sample, in order to estimate its orientation, new surface samples will be placed at a distance which will be larger or equal than the distance at which the furthest neighbor sample is placed.

One possible shape for the propagation region is a rectangle, although others could be used with similar result. Given a seed surface sample, we define two normal vectors $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$ which are orthogonal to each other and also to the normal $\hat{\mathbf{n}}$. By construction, these two vectors $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$ lie on the plane defined by the seed surface sample's position and orientation.

Rectangular propagation region. In order to generate this region, the following three random values are required for each sample:

1. Normal offset ω_n, or distance between the seed sample and the new random sample in the axis of the seed sample's normal, $\hat{\mathbf{n}}$.

2. Axial offset ω_u, or distance between the seed sample and the new random sample in the $\hat{\mathbf{u}}$ axis.

3. Axial offset ω_v, or distance between the seed sample and the new random sample in the $\hat{\mathbf{v}}$ axis.

Let ρ_n be a small maximum distance between surface samples such that these are close enough to assume they lie on the same plane. Let now ρ_r be a larger distance about one order of magnitude larger than ρ_n that is still small enough to assume a large number of surface samples are almost coplanar. Let n, u and v be uniform random variables in the range $[0, 1]$. Then, the previous random values ω_n,

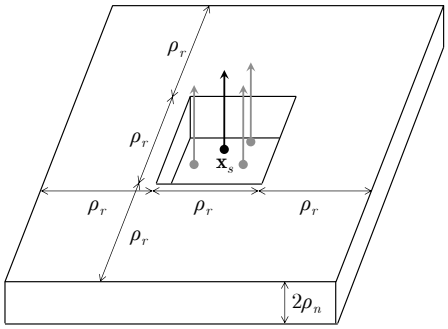


Figure 6.2: The rectangular propagation region around a surface sample $\mathbf{x}_s$ –and its neighbor samples, used for estimating the local orientation–. It favors propagation on the plane orthogonal to the normal vector while allowing small curvatures with variations of up $2\rho_n$ along the normal

ω_u and ω_v can be generated as follows:

$$\begin{aligned} \omega_n &:= \rho_n \left(2n - 1\right) \\ \omega_u &:= \begin{cases} \rho_r \left(u + \frac{1}{2}\right), u \geq 0 \\ \rho_r \left(u - \frac{1}{2}\right), u < 0 \end{cases} \\ \omega_v &:= \begin{cases} \rho_r \left(v + \frac{1}{2}\right), v \geq 0 \\ \rho_r \left(v - \frac{1}{2}\right), v < 0 \end{cases} \end{aligned} \tag{6.1}$$

A gap between a seed surface sample and the ones obtained from propagation is left empty in order to apply the propagation algorithm iteratively, at increasingly smaller scales, until the desired number of surface samples is reached. With this, the omni-directional search for new surface samples using the same propagation region at different scales can be performed without creating clusters of surface samples at small distances. Also, since any surface sample is surrounded by three additional surface samples at a small distance –used for estimating its orientation–, we leave an extra space $\rho_r/2$ in order to avoid the generation of larger clusters of samples in the vicinities of seed surface samples.

The final random sample is computed by combining the contributions of the three random values:

$$\mathbf{x}_r = \mathbf{x}_s + \omega_n \hat{\mathbf{n}} + \omega_u \hat{\mathbf{u}} + \omega_v \hat{\mathbf{v}}, \tag{6.2}$$

with $\mathbf{x}_r$ and $\mathbf{x}_s$ the random sample in a rectangular propagation region and the seed surface sample, respectively. The appearance of the rectangular propagation region is shown in Fig. 6.2. In the figure, a seed sample $\mathbf{x}_s$, surrounded by three neighbor samples used for estimating the local surface orientation, is the center of the oriented propagation region.

6.3.2 Iterative propagation algorithm

Let $q := \{\mathbf{x}_s\}$ be a queue of seed surface samples and N_s the target number of surface samples. Initially, q contains N_q seed surface samples obtained in the *scouting* stage (q does not contain the additional samples obtained for the estimation of the orientation, since propagating around them would generate local clusters of samples). ρ_n and ρ_r are initialized to valid values, according to the scene scale.

Then, while the required number of surface samples N_s is larger than the current number N_c –which is initially equal to $4N_q$, *i.e.* the total number of samples adding the seed surface samples and their neighbors–, the following algorithm is iteratively applied at monotonically decreasing scales (note step 6):

1. Create an empty queue of seed surface samples q'.
2. Pop the first seed sample $\mathbf{x}_s$ from q.
3. Obtain a test surface sample $\mathbf{x}_t$ lying in the rectangular propagation region and perform the foreground, contour and orientation tests introduced in the initialization stage. If $\mathbf{x}_t$ is valid (it belongs to the surface of the visual hull and can be correctly oriented), add it to q', store its neighbors in a separate list and update N_c.
4. Repeat the previous step until 4 new surface samples are obtained by propagation of $\mathbf{x}_s$.
5. Repeat from 2 to 4 until q is empty.
6. Set $q := q'$ and update the propagation region dimensions $\rho_n := \rho_n$ and $\rho_r := 0.5\rho_r$.

At the end, a total of N_s surface samples are obtained at a very small computational cost when compared to the initialization or *scouting* stage, as we will see in the experiments in Chapter 7. This results from the fact that the propagation region is designed to exploit the local distribution of surface samples.

Note that the additional gap introduced in the propagation region allows applying an omni-directional search for new surface samples at different scales of the propagation region without creating clusters of surface samples at small distances. This is illustrated in Fig. 6.3.

6.4 Efficient Sampling Schemes

Analyzing the strategy presented in this chapter, it is clear that the bottleneck resides in the initial search for seed surface samples, which is driven at random in

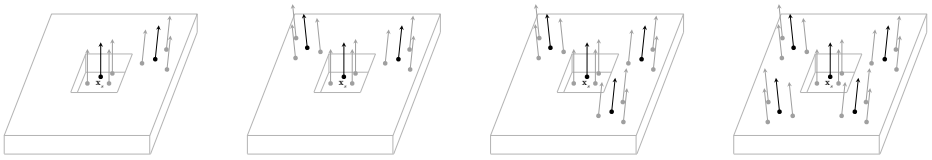


Figure 6.3: Propagation of a surface sample –and three neighbor samples– in order to generate 4 new surface samples –with three neighbor samples each–. Such propagation scheme allows to multiply by 5 the number of samples in every iteration, thus densely covering the surface

usually large working volumes –the space where people interact while being captures by the multi-camera rig–. Even though most of the surfaces samples are obtained during the propagation stage of the method, a large part of the computational resources are dedicated to this initial search.

In this section, two different techniques are proposed, which reduce the amount of computational resources spent in the initial search of surface samples. The first one is based on an idea already presented in Chapter 5, the existence of temporal correlations between the position of surfaces samples in a given a time instant and the next one, whereas the second one is based on considerations about the spatial sampling density implicit to the resolution of the input images.

6.4.1 Dynamic sampling

Similar to the dynamic surface sampling strategy presented in Section 5.3, the statistical approach for surface sampling can also benefit from temporal correlations in multi-view sequences. Indeed, the information about the position of surface samples in previous time instants can reduce the time required to find the seeds in the current time instant by limiting the volume in which new surface samples are to be searched for.

The approach taken in this chapter is not as tight as the one presented in the previous one. Indeed, only the seed samples are transferred from one time instant to the next one, leaving the complete covering of the surfaces to the efficient propagation scheme presented in Section 6.3. Thus, in this case an approximation of the dense velocity field of the surface samples cannot be obtained as a sample-wise first order estimate.

Assuming that the seed surface samples are evenly distributed over the surfaces of objects –which actually occurs, due to to the initial random search for surface points–, new positions for these samples can be safely searched for by independently setting a symmetric search region with a parametric *radius for dynamic search* ρ_d. This way, the placement of new seed samples for the current time instant does not favor any specific direction. Given three random values u, v, w in the range $[-1, 1]$,

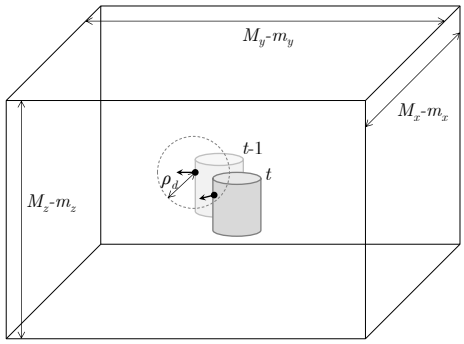


Figure 6.4: By exploiting temporal correlations between consecutive frames, the search space for seed surface samples can be reduced from the whole volume of $(M_x - m_x) \times (M_y - m_y) \times (M_z - m_z)$ to a smaller one $(4/3\pi\rho_d^3)$, thus increasing the search efficiency

such that $u^2 + v^2 + w^2 \leq 1$, the candidate to seed sample is computed as follows:

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \rho_d(u, v, w)^\top. \tag{6.3}$$

When the surface candidate and its three neighbor samples obtained for orientation estimation fulfill the three conditions presented in Section 6.2, they are added to the list of seed surface samples and the list of neighbor samples for the current time instant, respectively. Then, the next seed sample from the previous time instant is used for searching a new seed for the current time instant. This dynamic alternative to the initialization algorithm proceeds until finishing processing the whole list of seed surface samples from the previous time instant.

Fig. 6.4 illustrates the gain in search efficiency that can be achieved by exploiting the temporal correlation of the scene in this manner. Instead of finding a correct seed surface sample in the complete volume $V = (M_x - m_x) \times (M_y - m_y) \times (M_z - m_z)$, the search volume is reduced to $V' = 4/3\pi\rho_d^3$, which on its hand increases the probability of succeeding at finding a surface sample $\propto V/V'$.

After this alternative initialization, the propagation algorithm proceeds as in Section 6.3, resulting in the final sampling of the surface with the desired number of surface samples.

6.4.2 Multi-resolution sampling

The main purpose of this technique is to reduce the amount of random searches needed to find the initial seed samples by increasing their success probability. Please note that, given a set of silhouettes corresponding to different views of a scene, the likelihood of finding a contour pixel in random searches decreases when the resolution of the images increases. As a reminder, the projection onto a contour pixel for at least one silhouette is a requirement for a 3D point to be part of the

silhouette-consistent surface –see Section 6.2–.

Let $N_x \times N_y$ and l_c be the resolution of the input images –measured in pixels– and corresponding contour length –also measured in pixels–, respectively. If the resolution of an image is reduced by decimating in each dimension by a factor δ, the corresponding contour length is also divided by δ.

Now let a randomly chosen 3D point project onto one of the input views. The success probability at full resolution π_f can be considered proportional to $\frac{l_c}{N_x \times N_y}$, assuming that random points project with equal likelihood to any point in the image. Then, for the success probability after decimation, $\pi_d \propto \frac{l_c/\delta}{N_x/\delta \times N_y/\delta} = \delta \pi_f$, the result is that it increases with respect to that at full resolution. This translates in a reduced number of random attempts to find a surface point.

In order to accept a generated 3D random point $\mathbf{x}_l$ as a low-resolution seed to the silhouette-consistent surface, two conditions have to be fulfilled:

1. The 3D point projects onto a foreground pixel for all low-resolution views.
2. The 3D point projects to a contour pixel for at least one of these views.

The low-resolution seeds obtained with this method cannot be used as surface points because they are not accurate with respect to the high-resolution silhouettes. However, similar to the case of dynamic sampling, an actual surface sample can be found in a local neighborhood of low-resolution seed. This situation is illustrated in Fig. 6.5. The complete algorithm for multi-resolution sampling is as follows:

1. A list of low-resolution seeds is obtained from the decimated multi-view set.
2. As shown in Fig. 6.5, for each low-resolution seed, we apply an approach similar to that in the previous section, but using a search region around each seed with a *multi-resolution search radius* ρ_m, which should be as small as possible (in order to be efficient), but large enough to be able to find a valid surface sample in the region delimited by the back-projection of the pixels of the low-resolution views. As a result, a list of oriented seed samples and their corresponding neighbors is obtained.
3. Finally, the propagation procedure covers the entire surface with the desired number of surface samples.

6.5 Surface post-processing

In order to improve the visual appearance of the reconstructed surface, some refinements and addition of complementary information can be applied to the resulting

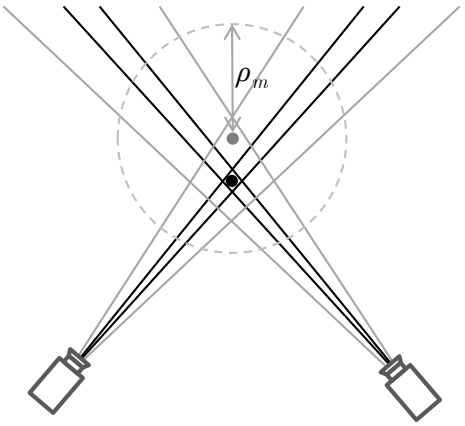


Figure 6.5: The search for a low-resolution seed –gray dot– needs a much smaller number of random searches to obtain than the high-resolution counterpart –black dot–. In order to obtain the high-resolution seed from the low-resolution one, a local search a region around the latter must be performed. The area between the gray and black half-lines starting at each camera correspond to the back-projection of a low-resolution contour pixel and a high-resolution one, respectively

set of surface samples. In the development of this thesis, three post-processing operations have been considered.

The first one is a finer estimation of the surface normals by using the distribution of surface samples in a wider area that comprises more samples than the used to obtain the surface orientation in the initialization stage, thus providing more robust estimates for the surface orientation. The second one is an anisotropic smoothing to obtain a smooth surface at a sup-pixel scale. The third and final one assigns a color to each surface sample without imposing photo-consistency, targeting real-time applications.

6.5.1 Fine normal estimation

Assuming the surface to have a large local curvature radius, the same least squares method presented in Section 4.4 can be applied for fitting a plane to a set of points in a neighborhood of each point on the surface offering better robustness than the one of the method for local estimation introduced above.

Let r_n be an arbitrary maximum distance for neighbor points satisfying the previous assumption. The algorithm applied to each surface point $\mathbf{x}_i$ works as follows:

1. Define an extrinsic neighborhood $\nu_i := \{\mathbf{x}_j\}$ including each surface point $\mathbf{x}_j$ at an Euclidean distance from $\mathbf{x}_i$ smaller or equal to r_n.

2. Estimate the mean position $\boldsymbol{\mu}_i$ of ν_i and modify the position of each neighbor point $\mathbf{x}_j$ by subtracting it $\mathbf{x}_j := \mathbf{x}_j - \boldsymbol{\mu}_i, \quad \forall \mathbf{x}_j \in \nu_i$.

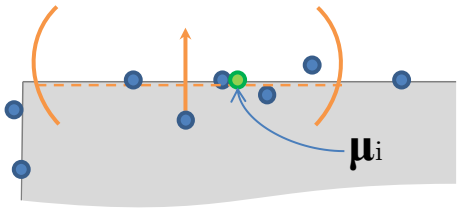


Figure 6.6: Normal estimated from a set of points by fitting a plane passing through the point resulting from averaging the positions of the complete set ($\boldsymbol{\mu}_i$)

3. Create a matrix $\mathbf{M}_i = \sum_{\mathbf{x}_j \in \{\nu_i\}} \mathbf{x}_j \mathbf{x}_j^\top$.

A vector $\hat{\mathbf{n}}_i$, orthogonal to the plane fitted by least squares to the neighborhood ν_i, is obtained as the eigenvector of largest eigenvalue of $\mathbf{M}_i$. Finally, the sign of $\hat{\mathbf{n}}_i$ has to be determined. Let λ_i be the average projection norm of the estimated normals $\hat{\mathbf{n}}_j$ of each $\mathbf{x}_j \in \nu_i$ onto $\hat{\mathbf{n}}_i$. If $\lambda_i > 0$, $\hat{\mathbf{n}}_i$ is considered to point outwards and therefore its sign remains unchanged whereas, if $\lambda_i < 0$, it is forced to point outwards by changing its sign: $\hat{\mathbf{n}}_i := -\hat{\mathbf{n}}_i$. The method is illustrated in Fig. 6.6.

6.5.2 Anisotropic smoothing

Once normals have been robustly obtained with the application of the previous method, the surface orientation can be used to eliminate the high-frequency noise in the placement of the surface samples due to the finite resolution of the input silhouettes. We plan to smooth surfaces along the direction of the normal of each sample. This technique allows neutralizing the effect of local skewness in the distribution of the samples in a neighborhood, while providing a suitable smoothing of the detected surface samples' positions.

Thus, in order to obtain a more visually appealing distribution of surface points, a spatial smoothness constraint is imposed to a local neighborhood of each surface sample. Let the superscript o denote an output sample and r_n a neighborhood radius, equal to the one used for normal estimation; then, the algorithm applied to each surface sample $\mathbf{x}_i$ works as follows:

1. Define a neighborhood $\nu_i := \{\mathbf{x}_j\}$ including each surface sample $\mathbf{x}_j$ at an Euclidean distance from $\mathbf{x}_i$ smaller or equal to r_n with a normal at an angle smaller than θ_ν with respect to the normal of the sample under consideration: $\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j > \theta_\nu$.

2. Estimate the mean position $\boldsymbol{\mu}_i$ of ν_i. Set the new value $\mathbf{x}_i^o := \mathbf{x}_i + \alpha \hat{\mathbf{n}}_i$, with $\alpha = (\boldsymbol{\mu}_i - \mathbf{x}_i) \cdot \hat{\mathbf{n}}_i$.

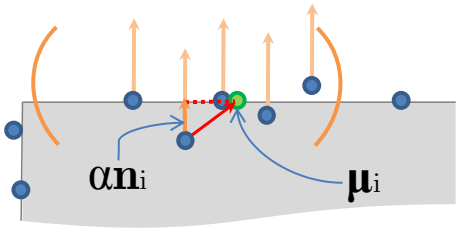


Figure 6.7: Each surface sample is displaced in the direction of its normal until intersecting the plane orthogonal to its normal, producing an anisotropic smoothing of the sampled surface

3. If $\mathbf{x}_i^o$ does not project onto any silhouette contour, repeat the previous step after setting $\alpha := \frac{4}{5}\alpha$.

Had we chosen to apply an isotropic smoothing technique, clusters in regions with higher sampling density and gaps in their emptier counterparts would have appeared. An illustration of an ideal case where the displacement results in a perfect placement of the surface sample –while projecting onto at least one silhouette contour– can be found in Fig. 6.7.

6.5.3 Surface coloring

Exploiting photo-consistency in this sampling approach would present the same type of difficulties as in the deformation-based approach from the previous chapter. The explicit computation of the visibility of each surface sample in order to impose global photo-consistency might be too costly for real-time applications, although a similar method to that in Section 5.4 could be applied to that respect.

However, in order to add color information to the reconstructed surface samples, a two-pass approach can be adopted, which will not provide photo-consistent surfaces but will contain important texture information that can be used in real-time applications. Such approach is based on the photo-consistency method in Section 5.4, but removing the possibility of changing the position of any surface point, thus exiting after the first iteration. Therefore, in first place a depth map for each view is obtained by projecting each surface point onto the corresponding viewpoint –with a sufficient area, in order to avoid hidden surface points to be visible through gaps between visible ones– and storing the closest depth value of each pixel.

In the second pass, we can determine the visibility of each surface point from each available view by projecting it one more time and comparing its depth to that of the corresponding pixel in the depth map. Using this information, we collect the colors of the 3 best oriented cameras where a surface point is visible. The choice of these 3 cameras is based on the angles between the surface point normal and the $-\hat{\mathbf{z}}_c$ vectors of each camera c's coordinate system expressed in world coordinates. For

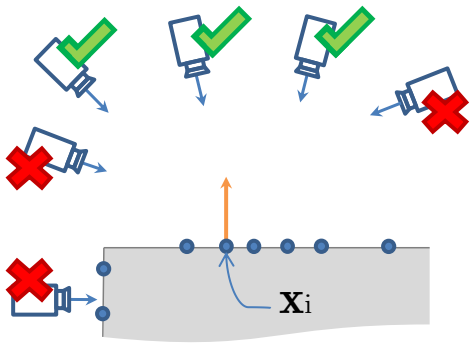


Figure 6.8: Given the normal (orange arrow) of a surface sample $\mathbf{x}_i$, its visibility from each view (computed using depth maps) and a set of cameras with orientation vectors $\hat{\mathbf{z}}_c$ (blue arrows), the three best oriented cameras viewing the sample can be selected

each camera, $-\hat{\mathbf{z}}_c$ is a vector pointing towards its center of projections –Appendix A– in a direction orthogonal to the image plane. Let $\hat{\mathbf{n}}_i$ be the normal of an oriented surface point. Then, $\omega_{i,c} = -\hat{\mathbf{z}}_c \cdot \hat{\mathbf{n}}_i$, is taken as orientation score of a camera c for the point $\mathbf{x}_i$. The selection of the three best oriented cameras for a given point is illustrated in Fig. 6.8. The final color of the surface point is assigned as a weighted average of the colors sampled from the three best oriented views, with the weight of camera c defined as $w_{i,c} = \omega_{i,c} / \sum_{c'=1..3} \omega_{i,c'}$.

6.6 Conclusions

In this chapter we have presented an alternative sampling strategy to obtain the position and orientation of samples of the surface of a visual hull that outperforms the other two sampling approaches presented in the previous chapters in terms of computational load and ability to scale to a large number of input views, as it will be seen in Chapter 7, where all the presented sampling strategies are validated.

This is accomplished thanks to an alternative search for surface samples driven by statistical principles that aim at exploiting the spatial correlation on the location of surface samples. The main advantage of this approach is that, although it initially generates a sparse surface sampling with a low search efficiency –random search–, an efficient search strategy in regions very likely containing surface samples around existing ones provides a fast dense sampling of surfaces. Among its advantages, this technique allows to define exactly the number of samples in the reconstructed surface. This feature improves the ability of this sampling strategy –compared to the one presented in the previous chapter– of providing surfaces that, once re-projected onto the original viewpoints, accurately describe the original images. Furthermore, the algorithm can also exploit correlations in the temporal axis. The inefficient initial search for a sparse sampling of the surface is not required from frame to

frame. Instead, a search over a suitably smaller range around each sample can be used. In order to improve the efficiency of the initial scouting stage when no information about previous time instants is available, a multi-resolution technique is also presented. The gains achieved with these two enhancements for the scouting stage (dynamic and multi-resolution) will also be presented in Chapter 7.

This technique still presents a limitation when compared to the image-based approach presented in Chapter 4: the application of photo-consistency constraints still requires costly global visibility computations, like in the approach presented in Chapter 5. However, in comparison to the technique presented in Chapter 5, this new algorithm is easy to parallelize, which makes it suitable to its application in GPU contexts. Furthermore, this sampling strategy is the fastest among the three presented in this thesis, showing the smallest computation time for a given number of surface samples, especially when considering larger multi-view settings, where the efficiency at exploiting spatial correlations of the proposed method shows up more clearly.

Chapter 7

Surface Sampling Results

In this chapter, the sampling methodology proposed in this first part of the thesis is validated by evaluating the results obtained when applying each of the techniques presented in the previous three chapters to different sets of multi-view static scenes and video sequences. Some of the results are purely qualitative, depicting the apparent improvement in quality of the proposed methodology with respect to some classical methods. A quantitative evaluation of the new techniques is also included, regarding different aspects relevant to their performance, such as the computational load and geometric or photometric accuracy.

This chapter is divided in four sections. The first one presents photo-consistent surfaces obtained by using the image-based sampling strategy introduced in Chapter 4. The next one presents silhouette-consistent surfaces obtained from both static multi-view scenes and dynamic multi-view sequences, using the surface deformation technique from Chapter 5. The third section presents colored, silhouette-consistent surfaces from multi-view sequences by using the methodology described in Chapter 6. Finally, a comparison of these three sampling strategies is presented.

7.1 Image-based Surface Sampling

The technique presented in Chapter 4 is able to reconstruct the visible surfaces of foreground objects from a very sparse set of images. An illustration of some of its stages is shown in Figure 7.1. It can be summarized as:

1. First, obtaining candidate depths for the image points in each view, based on photo-consistency measurements between pairs of pixels from neighbor cameras, using epipolar constraints.

Figure 7.1: Image-based surface sampling. From left to right: two sample input images out of six available; initial depth estimates for one of the input viewpoints; reconstructed surface points; reconstructed surface rendered with colored colored surface samples from two novel viewpoints

2. Then, surface points are extracted by iteratively carving away those points which are not globally photo-consistent.

3. Finally, a normal for each surface point is estimated by fitting a plane to a local neighborhood.

In the following, two different types of results are presented. The first one is a qualitative comparison between the results obtained with both typical small baselines that are considered in multi-view stereo scenarios and more challenging ones obtained by dramatically increasing the baseline –reducing the number of views– in the same scenes. These results are accompanied by examples of surface samples from static scenes in real-world scenarios with known –and removable– background in challenging wide-baseline configurations.

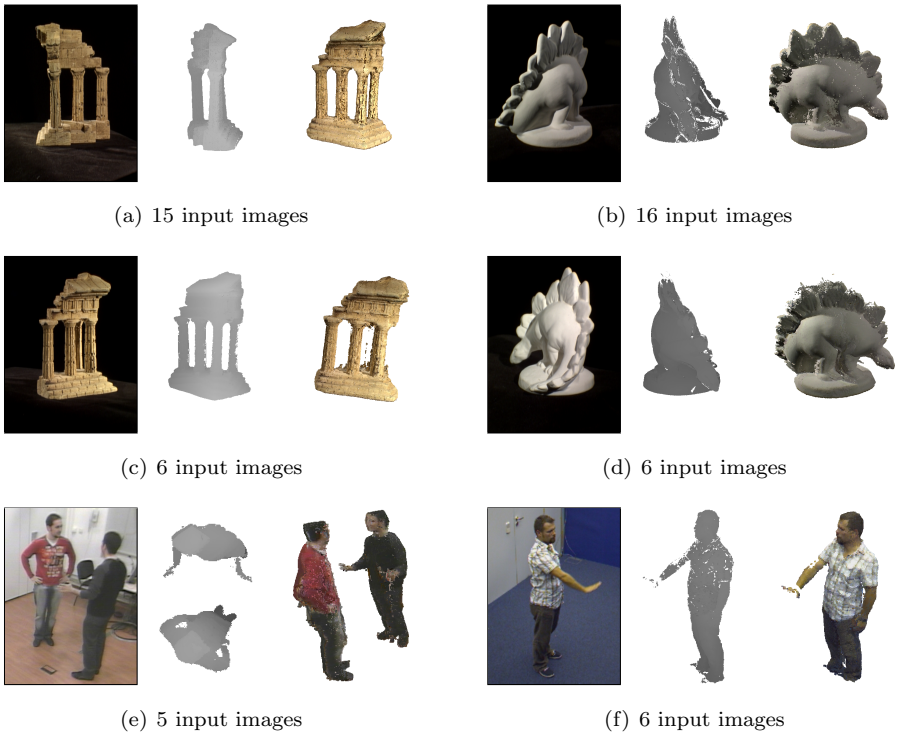
The second type of results, focusing on the target scenarios, present a quantitative measure of the quality improvement that can be achieved by a surface sampling strategy, compared to a volumetric sampling strategy, under equivalent conditions.

7.1.1 Qualitative validation

The datasets listed in Table 7.1 have been used as input in our experiments. In the table, apart from the number of available input images and their resolution, we have also included the maximum allowed squared RGB distance for photo-consistency –manually adjusted– that has been used for each dataset, ρ.

The first two datasets, *temple* and *dino*, are part of the benchmark promoted by [Seitz et al., 2006]. In the *temple* dataset, one of the 16 available images has been removed, because it was repeated. The datasets *dino-s* and *temple-s* are obtained by just picking 6 of the available views. This produces a very sparse scenario that helps us demonstrating how the sampling strategy is actually able to extract photo-consistent surfaces under such adverse conditions.

The *persons* dataset corresponds to 5 views of two persons captured at the *smart room* facility of the Signal Processing Group of the UPC, whereas the *shirt* dataset



(a) 15 input images (b) 16 input images

(c) 6 input images (d) 6 input images

(e) 5 input images (f) 6 input images

Figure 7.2: Sample results on the used datasets. (a) *temple*, (b) *dino*, (c) *temple-s*, (d) *dino-s*, (e) *persons* and (f) *shirt*. In each case, one of the input images is shown, along with the initial estimate of one of the depth maps corresponding to one of the input views and the reconstructed surface samples seen from a novel viewpoint

consists of 6 views of a person in a smart room environment in the *Telefonica R&D* facilities in Barcelona.

The only relevant parameter for obtaining correct reconstructions in all the scenarios is the aforementioned photo-consistency threshold ρ. For example, in datasets where the color captured by different cameras has not been calibrated, as in *persons*, this threshold must be loose enough to allow the reconstruction of almost complete surfaces. Please note that the higher the threshold, the looser, since our photo-consistency test (Section 4.2) measures Euclidean color distances. Whereas a high threshold implies the acceptance of some outliers as surface elements, for us it is more important to obtain an almost complete surface, with both visualization and analysis applications in mind.

The results obtained from testing our algorithm with the datatsets in Table 7.1 are shown in Fig. 7.2. For each dataset, we have included one of the input views, an initial depth estimate for one of the viewpoints and the resulting surface rendered with colored surface samples viewed from a novel viewpoint.

The method shows a relatively fast performance, mainly when compared to

Name	Views	Image Size	ρ
temple	15	640×480	100
dino	16	640×480	30
temple-s	6	640×480	100
dino-s	6	640×480	30
persons	5	768×576	3000
shirt	6	640×480	1000

Table 7.1: Datasets for the qualitative validation of the image-based sampling approach. The large *temple* and *dino* datasets are included for reference. The only parameter that needs tuning is the photo-consistency threshold ρ over the squared Euclidean RGB distance

multi-view stereo techniques, with computation times ranging from more than 3 minutes for the complete *dino* dataset (16 views) down to a few seconds for the *persons* dataset. These times are obtained on an Intel Xeon CPU at 3.0 GHz.

The reconstructions with the *temple* and *dino* datasets are useful for comparative purposes. The very sparse datasets (those with less than 10 input viewpoints placed all around the scene), which are prohibitive for multi-view stereo techniques due to the wide baseline, can be handled with the proposed sampling strategy. Indeed, it trades-off geometric or photometric accuracy with spatial coverage of the scene.

7.1.2 Quantitative results

The neighbors of each view are automatically obtained by taking the 3 closest viewpoints. The proximity of two viewpoints is defined as $\pi_{ij} = \pi_{ji} = \left(1 - \frac{1}{2}\cos(\alpha_{ij})\right)\delta_{ij}$, where α_{ij} is the angle between the two unitary vectors orthogonal to the image planes i and j and δ_{ij} is the Euclidean distance between the centers of projections of the two views. Photo-consistency thresholds have been adjusted empirically.

The dataset used for the evaluation is composed by 33 static multi-view scenes captured in a controlled environment by a set of 18 cameras placed around the scene of which we will make subsets for the experiments. Eight of these cameras offer partial views of the objects of interest in the scene, whereas two other cameras are reserved as ground-truth for indirect evaluation of the reconstruction quality. The datasets for evaluation are obtained by picking some of the views for reconstruction. As a result, three datasets with 6, 8 and 16 cameras respectively are available for our experiments. The corresponding foreground silhouettes are automatically extracted and show misclassified pixels. The foreground silhouettes of the two ground-truth views are manually extracted.

The evaluation procedure consists in using the 6, 8 or 16 available views for reconstructing the photo-consistent shapes of the foreground elements of the scene,

	F-measure		PSNR	
# Views	Surface patch	Voxel	Surface patch	Voxel
6	0.85	0.84	40.64 dB	34.77 dB
8	0.87	0.86	41.67 dB	37.08 dB
16	0.88	0.88	43.62 dB	39.94 dB

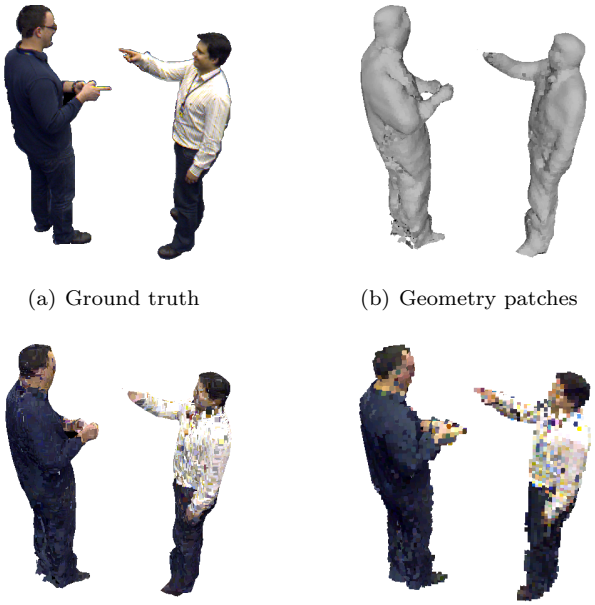
Table 7.2: Evaluation of image-based sampling and volumetric photo-consistent reconstruction schemes considering a growing number of input views for the experiments illustrated if Fig. 7.3

projecting them onto the two unused viewpoints and extracting the performance measurements from comparison with the ground-truth. The measured magnitudes are:

- *F-measure*, or the harmonic mean of *precision* (rejection of false foreground pixels) and *recall* (rejection of missing foreground pixels), measured on the silhouettes,
- *PSNR* of the pixels that are part of the foreground on both the ground-truth and testing silhouettes.

A volumetric equivalent of the proposed method has been implemented, which also uses the proposed photo-consistency test and the visual hull-bounding of the final photo-consistent reconstructed shapes. In order to compare the results, we follow the equivalency condition between voxels and image sampling density proposed in a previous work [Casas and Salvador, 2006]. Thus, given a sampling distance in each major direction x and y used on the available input images, $\sigma_{\{x,y\}} = 4$ pixels in our experiments, we define its equivalent voxel size, measured in units of the coordinate system of the scene.

The average results obtained by comparing the projections of the reconstructed shapes with the two ground-truth images for the surface patch-based and voxelized methods for each of the three considered datasets are shown in Table 7.2. As it can be observed, the accuracy of the proposed method is higher than that obtained by a voxelized approach, both in terms of shape (F-measure) and color (PSNR). Regarding the latter, our measurements confirm that surface color is not properly represented by the inaccurate surface description provided by a volumetric reconstruction. When analyzing the results, please keep in mind that the ground-truth images have not been used for reconstruction. Some results for the evaluated scenes are shown in Fig. 7.3, including geometric and textured renderings of the reconstructed shapes from novel viewpoints. The computation time of the proposed surface patch-based method is within one order of magnitude larger than that of its volumetric counterpart (ranging from fractions of a second for the datasets with 6 views up to more than ten seconds for the datasets with 16 views).



(a) Ground truth (b) Geometry patches

(c) Colored patches (d) Voxels

Figure 7.3: Reconstruction results for 8 cameras. The groundtruth image has not been used for reconstruction. (a) Groundtruth image masked with its manually extracted silhouette. (b) Reconstructed geometry represented by surface patches. (c) Colored surface patches. (d) Voxelized colored surfaces with equivalent resolution

7.2 Surface Sampling by Deformation

The methodology for extracting silhouette-consistent surface samples by the deformation of an existing surface is presented in Chapter 5. As an advantage with respect to the image-based approach, it allows the exploitation of temporal redundancies, following a tracking strategy. As usual among tracking strategies, two separate stages provide initialization and tracking:

- *Initialization: Static reconstruction.* First, samples of a silhouette-consistent surface are obtained for an initial time instant –without having any knowledge of the shape at previous time instants– by exploiting only spatial redundancies.
- *Tracking: Dynamic reconstruction.* Then, the knowledge of the shape in the previous time instant allows for a more efficient sampling of the surfaces in the next time instant, by exploiting both spatial and temporal redundancies.

Three experiments are used to validate the proposed method. The first experiment checks the consistency of the shapes resulting from dynamic reconstruction, compared to their equivalents obtained by static reconstruction, in order to validate the faster former method. The second experiment evaluates the usage of computational resources of dynamic reconstruction when compared to a classical volumetric

Sequence	Time static	Time dynamic	RMS distance
walking	3.43 s	1.07 s	0.000102
open arms	3.77 s	1.17 s	0.000134
move arm	3.45 s	0.93 s	0.000100
kick	3.45 s	1.23 s	0.000094
jump	3.41 s	1.16 s	0.000119

Table 7.3: Average RMS Hausdorff distance of surfaces obtained by dynamic reconstruction *vs.* static reconstruction. The maximal edge length of the associated bounding box is 0.101 in all cases

shape-from-silhouette algorithm. The third experiment quantitatively evaluates the accuracy of the method, using a synthetic scene for which we have 3D ground-truth data.

The synthetic scene is obtained as the projection of an animated human-like 3D model onto a number of virtual viewpoints. Therefore, the reconstructed surfaces can be compared to those of the actual model in every time instant.

For the other experiments, five real sequences with known background have been used, with 50 frames each. The silhouettes of a person captured by 8 cameras placed all around are extracted using [Stauffer and Grimson, 1999]. These sequences –*walking*, *open arms*, *move arm*, *kick* and *jump*–, show different types of motion, ranging from fast motion of limbs to slower displacement of the person.

The initial surface for the tracking strategy is defined by 10^5 surface points randomly distributed over a box-shaped surface with dimensions slightly larger than that of the bounding box of each scene; the diameter for normal estimation $2 \times r_n$ is set to 1/100 of its main diagonal and the expansion distance is set to $r_t := 5 \times r_n$. In order to reconstruct shapes when parts of the scene are not included in the frusta of all views, a minimal inclusion in 4 frusta is required. All measurements have been obtained from an Intel Xeon 3.0 GHz.

7.2.1 Static vs. dynamic reconstruction

Fig. 7.4 shows the reconstructed surfaces with color-encoded estimated velocities for their samples. The sequences *walking*, *open arms* and *jump* show displacement of large surfaces in scenes with displacement of large surfaces, whereas *move arm* and *kick* show fast motion of limbs in scenes with displacement of small surfaces. The velocity estimated by the dynamic reconstruction method shows outliers in areas where surfaces collide, but, overall, it plausibly describes the apparent surface motion.

In order to quantitatively validate the dynamic method as an efficient alternative

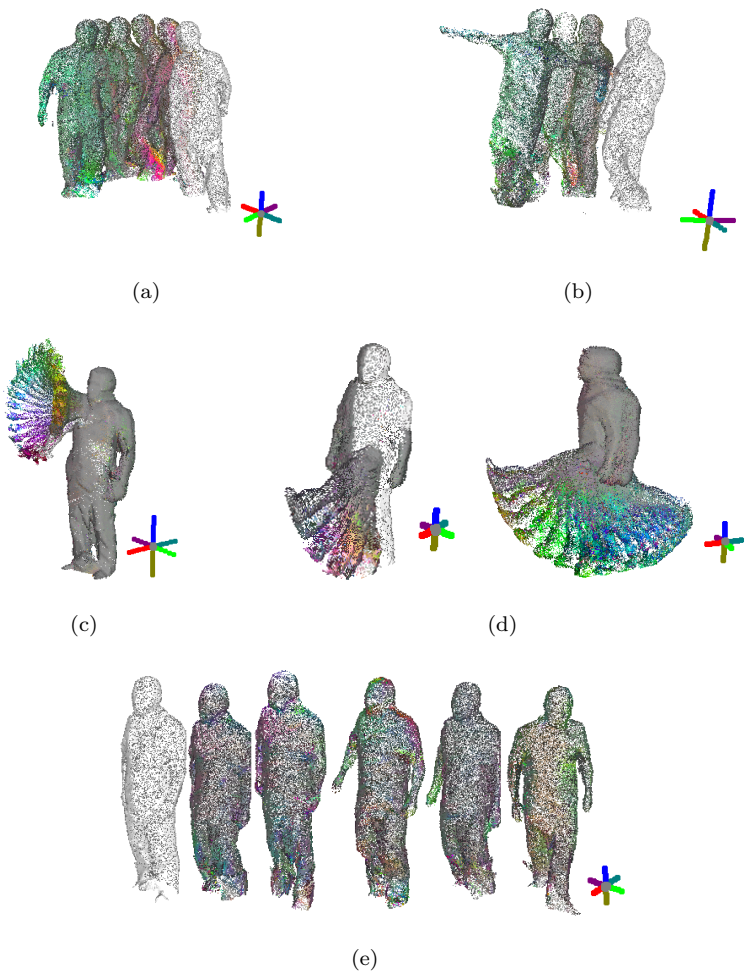


Figure 7.4: Surface samples from sampling by surface deformation and color-encoded velocity estimates for (a) *walking*, (b) *open arms*, (c) *move arm*, (d) *kick* and (e) *jump*. The RGB color code for velocity follows the included axes, although the normal information included for better visualization of shape under illumination slightly distorts its actual intensity. White colored points correspond to surfaces at initial time instants (static reconstruction)

to static reconstruction, each of the scenes have been processed with both methods, and the average RMS Hausdorff distance [Aspert et al., 2002] is measured between two meshes obtained by applying *Poisson reconstruction* [Kazhdan et al., 2006] on the sets of oriented surface points. The results in Table 7.3 reflect that shapes are practically equivalent and the processing times are clearly and consistently reduced.

7.2.2 Usage of resources

In Table 7.4, the memory usage for two configurations with approximately equal processing times shows how the reconstruction of surfaces with the proposed method needs less computational resources than the voxelized alternative and provides a

Method	Initial pts	Surface pts	Memory
Voxel	$121 \times 300 \times 311$	53706	700 MB
Dynamic	5×10^5	84962	495 MB

Table 7.4: Memory usage of surface sampling by deformation compared to that of volumetric shape-from-silhouette with approximate equal processing times of 2.2 s and 2.3 s respectively

higher level of detail. This can be visually confirmed in Fig. 7.5, where triangular meshes have been extracted from both voxelized shape-from-silhouette and the proposed method with *marching cubes* [Lorensen and Cline, 1987] and *Poisson reconstruction*, respectively. The memory usage and execution times in the voxelized approach are influenced by the implementation strategy of pre-computing the projections of all voxel centers and storing the pixel coordinates in *Look-Up Tables*, which greatly shortens the processing time.

7.2.3 Accuracy

Synthetic sets of silhouette sequences are generated by the projections onto a set of views of an animated mesh model. We utilize the average RMS Hausdorff distance between a *target mesh*, which is obtained either from voxelized shape-from-silhouette followed by marching cubes or from dynamic reconstruction followed by Poisson reconstruction; and a *sampled mesh*, which is the initial mesh model. The results are summarized in Table 7.5. As expected, dynamic reconstruction outperforms the voxelized shape-from-silhouette method in terms of accuracy, because the 3D sampling at regular positions that takes place in the voxelized method cannot accurately describe surfaces, and it also confirms that the dynamic reconstruction of shape and motion achieves a correct estimate of the visual hull from frame to frame, as suggested by the first experiment.

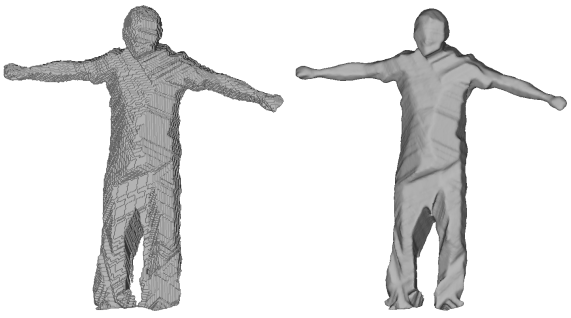


Figure 7.5: Comparison of the reconstructed surface with voxelized shape-from-silhouette (left) and the proposed method (right) for approximately equal processing times of 2.3 s and 2.2 s, respectively

Sequence	RMS dist. volumetric	RMS dist. dynamic
walking	0.001286	0.000664
open arms	0.000902	0.000447
move arm	0.000873	0.000587
kick	0.001100	0.000745
jump	0.000910	0.000517

Table 7.5: Average RMS Hausdorff distance for approximately equal processing times using volumetric shape-from-silhouette and dynamic sampling by deformation. The maximal edge length of the associated bounding box is 0.101 in all cases

7.3 Statistical Surface Sampling

The statistical surface sampling strategy presented in Chapter 6 is designed to efficiently obtain silhouette-consistent surface samples by directing the search to regions where a high likelihood of finding surfaces exists. This is achieved by a two-staged approach:

- First, an initial *scouting* procedure in the working volume, obtains a small amount of *seed* surface samples, which are costly to find but constitute a small percentage of the final set of surface samples.
- Then, a propagation strategy redirects the search to suitably defined regions around each seed sample, and quickly covers the complete surface until reaching the desired number of surface samples.

The bottleneck of the algorithm is the initial scouting, which takes the most of the computation time, due to the blind search that takes place in a large volume. Therefore, two strategies for reducing the computation time required to obtain the seed surface samples have been proposed:

- The first one, *dynamic* statistical sampling, exploits temporal correlations in multi-view sequences, which translates in a reduced search region for seed samples from frame to frame and posterior propagation.
- The second strategy introduces a *multi-resolution* technique, which performs a fast greedy search for surface points from heavily decimated input that is followed by a search for seed surface samples in small regions around each low-resolution surface point in a manner similar to that in the dynamic strategy.

The algorithm is parameterized by the number of output surface samples –kept constant as 100000 in our experiments– and the size of the different search regions used at every stage. All of these parameters, including those only used for the

Parameter	Value
Bounding box volume	$s_x \times s_y \times s_z$
Fine normal estimation radius ρ_n	$\frac{1}{200}\sqrt{s_x^2 + s_y^2 + s_z^2}$
Initial propagation radiusρ_p	$2\rho_n$
Local normal estimation radius ρ_l	$\frac{1}{2}\rho_n$
Dynamic scouting radius ρ_d	$8\rho_n$
Multi-resolution scouting radius ρ_m	$2\rho_n$
Number of samples N_s	100000

Table 7.6: Parameters for the statistical surface sampling, including those for dynamic and multi-resolution sampling. Note that the number of surface samples is restricted to 100000

improvement of the *scouting* stage using one of the two proposed alternatives, have been empirically determined in order to provide the expected behavior and are listed in Table 7.6.

For the experiments, three multi-view real sequences from datasets available in [4D Repository, 2010] and a synthetic one –*kung-fu girl* [Kung-Fu Girl, 2005]– have been used, which are listed in Table 7.7. As it is shown, two of the real sequences are captured by 16 views placed all around the scene, whereas the remaining one contains only captures from 8 views. The synthetic sequence is captured by 25 virtual cameras placed on a hemisphere around the animated model.

Three experiments have been performed. The first two experiments demonstrate the efficiency increase that can be achieved by applying any of the two types of proposed improvements to the initial scouting for seed surface samples, e.g., the dynamic sampling and the multi-resolution sampling. The third experiment shows a property about the scaling of the proposed surface sampling strategy to large multi-view settings that makes it ideal to be used in such conditions. All the results are obtained from a single-threaded implementation running on an Intel Core 2 Duo 2.8 GHz processor.

Sequence	# Views	# Frames	Resolution
dancer	8	201	780×582
children	16	339	1624×1224
martial	16	210	1624×1224
kung-fu girl	25	200	320×240

Table 7.7: Three multi-view real sequences from the 4D repository and a synthetic sequence (*kung-fu girl*) used for the experiments related to statistical sampling

Sequence	Default	Dynamic
dancer	1.02 s	0.46 s
children	2.73 s	0.92 s
martial	2.35 s	0.88 s
kung-fu girl	1.4 s	0.56 s

Table 7.8: Per-frame computation time with default and dynamic scouting for statistical sampling

7.3.1 Multi-resolution and dynamic sampling

In this section we present two experiments that demonstrate the increase in efficiency that can be achieved when modifying the initial scouting stage with the two proposed improvement types.

In Section 6.4.1, a dynamic scouting strategy has been presented, which reduces the computation time in multi-view video sequences by exploiting temporal correlation. This technique has been used for the four sequences. The average computation times for each sequence with and without dynamic scouting are listed in Table 7.8. Inter-frame similarity of the position of the surfaces of foreground objects can be used for reducing the search region for seed surface samples, which results in a clear increase in frame-rate.

In Section 6.4.2, a multi-resolution approach has been proposed to improve the efficiency of the scouting stage. The most important adjustable parameter is the decimation level in the input images. In Fig. 7.6, the execution times for sampling with different levels of initial decimation for the input silhouettes are shown. Note that the case with decimation set to one corresponds to the default scouting strategy.

The resulting efficiency gain obtained by both dynamic and multi-resolution

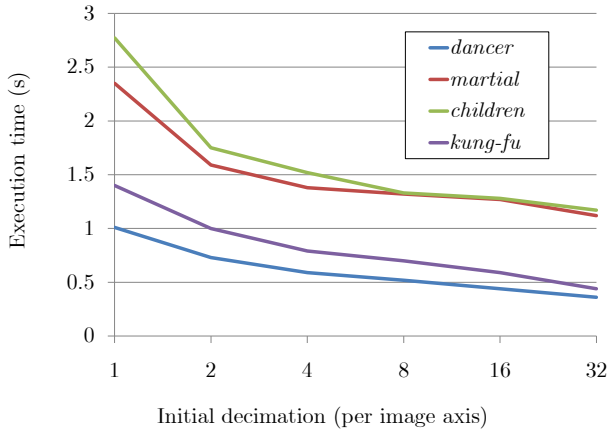


Figure 7.6: Reconstruction time *vs.* initial level of image decimation in multi-resolution scouting. A decimation of 1 is equivalent to the default scouting with full-resolution images

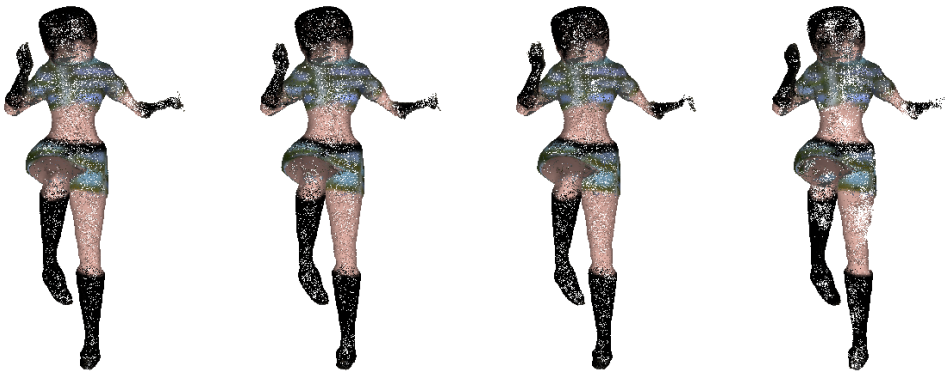


Figure 7.7: From left to right, statistical sampling results for the *kung-fu girl* sequence for initial decimation of 1, 2, 8 and 32 in each image dimension. Decimation of 1 is equivalent to default scouting. Incorrect values for the search radius lead to uneven distribution of seed surface samples, which cannot be correctly balanced during the propagation stage

scouting is similar. The extra cost from initially obtaining seed samples at low resolution in multi-resolution scouting is partly balanced with the fact that the low-resolution seeds are in average closer to the actual surface than the high-resolution seeds from a previous time instant in dynamic scouting, which translates in the choice of a multi-resolution scouting radius smaller than its dynamic scouting counterpart.

In Fig. 7.7, surface samples corresponding to a frame from the *kung-fu girl* sequence are shown, as obtained when using the multi-resolution scouting strategy with different levels of initial decimation. Due to an incorrect choice of the search radius for high-resolution seed surface samples, most of the low-resolution samples are discarded by the impossibility to find a high-resolution matching sample. The propagation mechanism is not designed to correct such situation, resulting in a highly uneven distribution of samples.

7.3.2 Efficiency in multi-camera environments

A limiting problem when working in multi-camera environments is the cost associated to the addition of each new camera. Typically, algorithms exploiting multi-view data can grow linearly –i.e., in volumetric reconstruction techniques– or even exponentially –i.e., in image-based approaches where multiple cross-validations between images take place– with the number of available cameras.

However, the proposed strategy for sampling benefits from the fact that the ratio between contour pixels and total number of pixels remains practically constant with respect to the number of available cameras when these are placed at similar distances from the objects of interest. Even better, the ratio between silhouette contour pixels

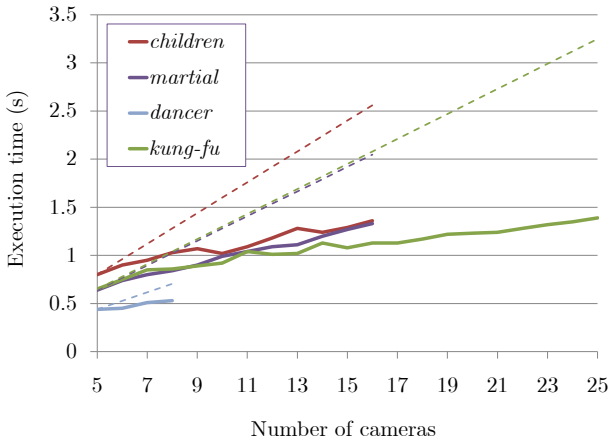


Figure 7.8: Reconstruction time *vs.* number of cameras for three real sequences from [4D Repository, 2010] and a continuous sequence from [Kung-Fu Girl, 2005] (all in continuous lines). The dash lines show the computation time that would be required for sampling if the computational cost of adding a view was constant (as in volumetric shape-from-silhouette). The details of the sequences are in Table 7.7

and occupied volume increases with the addition of each new camera. Thus, even if the cost of a projection test grows linearly with the number of views, the probability that a 3D point is a surface point also grows, partly balancing this effect. As shown by the following experiment, the result is that the proposed sampling strategy is well suited to large multi-camera settings.

Fig. 7.8 shows, in dashed lines, the extrapolated computation time from the first five up to the whole range of available views. As shown, the actual computation time grows at a much smaller rate for subsequent views, due to the introduction of a larger number of silhouette contour pixels. In Figs. 7.9 and 7.10, the reconstructed surfaces for one time instant of each sequence with increasing numbers of input views are shown as a set of surface samples or its interpolating mesh obtained with the algorithm presented in Part II, respectively. The marginal increase of detail in the resulting surface becomes smaller, and so does the marginal cost of adding each additional view. This shows the efficiency of the proposed sampling strategy.

7.4 Comparison of the Three Sampling Approaches

The three sampling strategies presented in this part of the thesis are now compared in terms of computation time and visual appearance of the resulting surface. The video sequence used for this comparison is *dancer*, from [4D Repository, 2010], listed in Table 7.7.

In Table 7.9, the average computation time per frame using each of the proposed techniques is listed. For the sampling strategy based on the deformation of an

Figure 7.9: Statistical sampling with increasing number of views. The leftmost column shows surface samples reconstructed from only 5 viewpoints of each sequence. Towards the right, surface samples obtained with increasing number of views. The rightmost column shows the surface samples reconstructed from the whole multi-view set for each sequence

Figure 7.10: Statistical sampling with increasing number of views. The continuous surface is obtained with the algorithm presented in Part II. The leftmost column shows surfaces reconstructed from only 5 viewpoints of each sequence. Towards the right, surfaces obtained with increasing number of views. The rightmost column shows the surfaces reconstructed from the whole multi-view set for each sequence

Method	No. of samples	Time	Initialization time
Image-based	56502	167.34 s	-
Deformation	56610	2.5 s	8.62 s
Statistical	56610	0.25 s	0.63 s

Table 7.9: Average computation time per frame (in seconds) for sequence *dancer* using the three sampling strategies with approximately equal number of reconstructed surface samples

existing surface, the obtained time corresponds to that of the tracking strategy (initialization 8.62 s). For the statistical sampling strategy, the time in the table corresponds to the application of the dynamic improvement in the *scouting* stage (initialization without multi-resolution decimation 0.63 s). The number of surface samples has been determined by the image-based strategy, with the deformation-based strategy configured in order to obtain a value as close as possible to the former and the statistical sampling configured in order to retrieve exactly the same number of samples as the highest one among the other two strategies.

The average computations times confirm the expected behavior of the three strategies. Whereas the image-based approach does not exploit any sort of spatial or temporal correlation on the distribution of surface samples and the strategy based on the deformation of an existing surface exploits mainly the temporal correlation (and indirectly also spatial during the regularization algorithm), the statistical sampling strategy is the one that best exploits these two types of correlations outperforming the other two methods in terms of computation time for a certain sampling density.

Fig. 7.11 shows the continuous surface corresponding to one time instant of the *dancer* sequence, obtained using *ball pivoting* [Bernardini et al., 1999] over the set of surface samples reconstructed with each of the three sampling strategies. Whereas the surface obtained with the image-based strategy is the closest to the actual surface –see the narrower head, achieved by imposing photo-consistency–, it also presents holes –like those on the chin of the dancer–. The surface obtained from the deformation-based strategy is practically equivalent to that obtained with statistical sampling, with a clearly higher presence of high-frequency detail and, most prominently, noise on the distribution of surface samples in the former, avoided by the application of anisotropic smoothing (Section 6.5.2 in Chapter 6) as post-processing in the latter.

In general, considering the processing time and the resulting topological correctness of the statistical sampling strategy, the presented results assess this strategy as the best of the three proposed strategies in terms of the goals of this thesis. Indeed, it is the approach that takes a smaller processing time –close to real-time operation using a single core of the CPU, while being easy to parallelize– and provides a closed surface, free of holes. Whereas the posterior application of *ball pivoting*, in

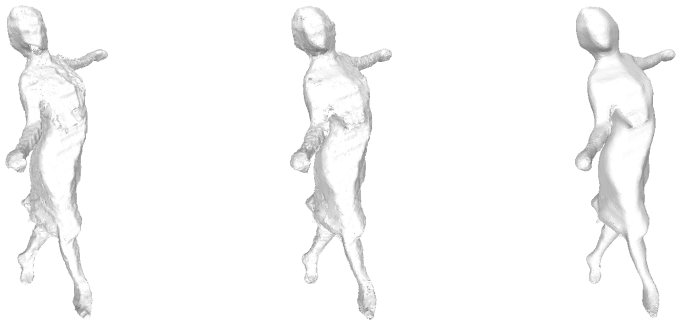


Image-based Deformation Statistical

Figure 7.11: Continuous surface for a time instant in the video sequence *dancer*, obtained by applying *ball pivoting* on the set of approximately 56000 surface samples reconstructed by the image-based, the surface deformation and the statistical sampling strategies

order to provide a continuous triangulated surface, renders this approach useful for off-line processing only, the surface interpolation algorithm presented in Part II of this thesis tackles the problem of tessellating a continuous surface between samples targeting speed and topological correctness.

7.5 Conclusions

The experiments presented in this chapter provide evidence of the high accuracy level that can be achieved by obtaining surface reconstructions using the techniques introduced in the previous chapters of this first part of the thesis. Different types of experiments have been conducted using each of the three presented surface sampling approaches, as well as a final comparison of the three methods.

Using the image-based surface sampling approach, experimental results show that, in order to obtain an equivalent resolution level in a volumetric methodology, a much higher amount of computational resources would be necessary. Compared to multi-view stereo methods, the photo-consistency test proposed in Chapter 4 proves useful in arbitrarily wide-baseline scenarios. This technique is the slowest among the presented ones, but it can be used to provide photo-consistent reconstructions when a small number of views is available.

With the surface sampling strategy by surface deformation, experiments show that the method allows taking advantage of temporal redundancies in dynamic scenes, which are the source of a great reduction of the required computation time and can be used to provide an explicit description of motion that can be used by higher level algorithms as an additional set of features detected from the raw data, as introduced in Chapter 5. This technique is much faster than the previous one,

due to the exploitation of temporal correlation, and can be, therefore, used with a larger number of views if the latency due to the slower initialization stage can be allowed in a specific real-time application.

Finally, using the statistical sampling strategy presented in Chapter 6, experiments show that it is highly efficient with respect to large number of input views, yet capable of obtaining reconstructions in very wide-baseline configurations, thanks to the proper exploitation of spatial correlations. Furthermore, the exploitation of temporal correlations is also introduced as a mechanism to improve the initial scouting stage for seed surface samples. Similarly, a multi-resolution technique can also be designed, which produces similar gains in processing time when no information from the current position of surfaces is available. When a large number of views is to be processed for real-time applications, this would be the algorithm that best fits the goals of the thesis presented in Chapter 1, thanks to the more principled exploitation of both spatial and temporal correlations.

As the final comparison between the three strategies shows, the statistical sampling approach is the fastest, resulting from the better exploitation of spatial correlation, and provides a topologically correct, closed surface. Due to the high degree of parallelization of the latter algorithm, it is also the best suited for GPU implementations. Indeed, a Master's Thesis developed by Marc Maceira and co-directed by my advisor and myself has recently resulted in a real-time implementation with speed-ups of around 5×–7×, with room for further improvements.

Part II

Surface Interpolation

Surface Interpolation

Whereas the previous part of the thesis copes with the definition of strategies for the reconstruction of sets of surface samples corresponding to foreground objects in multi-view video sequences, this second part tackles the problem of retrieving a closed, continuous surface –presented as a triangle mesh– in order to interpolate the information contained in such surface samples.

Again, the design of the corresponding methodology is driven by the goal of obtaining fast methods applicable in interactive applications, while offering topological guarantees in order to keep the resulting continuous surfaces usable for forthcoming visualization and multi-view analysis applications.

In the two chapters composing this second part of the thesis, we first present the details of the proposed method for surface interpolation –meshing algorithm– and, then, we compare its performance to that of other meshing methods in the literature, by using well-known datasets of surface samples obtained from range data. The application of the method proposed in this second part of the thesis on the sets of surface samples obtained by the best sampling approach in the first part is left for Part III, where a validation for the complete surface reconstruction approach –including both sampling and interpolation– is presented.

Chapter 8

Surface Interpolation

A set of surface samples constitutes a minimal discrete representation of a surface. In order to obtain a complete reconstruction of the surface, an interpolation scheme must be provided, which tessellates a continuous surface between its samples. Due to the possible irregularity resulting from the sampling scheme, interpolating a surface between these samples is a non-trivial problem that requires special attention, as it will be explained below. In this chapter, a novel meshing algorithm is introduced, which produces a two-manifold mesh from a set of surface samples. Such a mesh is useful for visualization purposes, but also for *analysis-by-synthesis* methods that might exploit convenient topological properties of 2D manifolds in 3D space.

8.1 Motivation

Surface samples can be used for visualization, without the need for an explicit computation of a continuous surface out of the rendering pipeline, as it has been demonstrated with impressing results in [Pfister et al., 2000] or [Guennebaud et al., 2004]. However, a discrete representation of a surface does not provide a complete description of its topology, which is usually required when processing 3D data (in some applications through surface parameterization [Hormann et al., 2007]). Furthermore, a suitable continuous description of the surface can also simplify visualization by using the standard rendering pipelines in current graphics hardware.

As mentioned above, the problem of interpolating a continuous surface given a set of discrete surface samples presents several challenges. In first place, the problem can be computationally expensive when the number of surface samples is large. The problem is also ill-conditioned: in general, some assumptions about the

topology of the surface have to be made in order to obtain a solution, which will in general not be unique. The combination of these problems translate to the main challenge, which is the definition of the ordering of the surface samples for surface interpolation. Furthermore, in order to be convenient for analysis and visualization, the interpolated continuous surface must guarantee certain topological properties, such as being closed and orientable.

8.1.1 Interpolation

Mathematically, interpolation can be described as a method of constructing new data points within the spatial range of a discrete set of known data points. It can also be interpreted as a specific case of curve fitting, in which the resulting curve goes exactly through the data points or samples.

In datasets where the independent variable –sample points x_i– is unidimensional, the order of the samples is uniquely defined. Thus, the problem in such case is just that of defining the *interpolant* function.

However, in datasets like the ones we deal with, the independent variable is a position in 3D space. Therefore, the order of samples, which will be used in order to generate the interpolant function, cannot be uniquely chosen, since a well-defined ordering of the samples does not exist. Indeed, the most challenging part of surface interpolation is precisely the definition of the neighborhood of each sample point, which in turn results in the explicit description of the surface topology. Whereas for analysis applications it is convenient to obtain a topologically correct continuous surface, obtaining such a surface in general introduces more complexity to the procedure.

The problem of defining the topology of the independent variable is therefore the main topic of this chapter. In the following, we justify the selection of an interpolant function by studying the different available interpolation types:

- *Polynomial.* The interpolant is a polynomial of a certain degree.
- *Spline.* The interpolant is a special type of picewise polynomial, or *spline.*

Polynomial interpolation

Given a set of $n+1$ data points (x_i, y_i) where no two x_i are the same –in case of a surface, x_i would be the coordinate of vertex i and y_i some attribute related to such vertex, i.e., its color–, one is looking for a polynomial p of degree at most n with the property

$$p(x_i) = y_i, \forall i \in \{0, 1, ..., n\} \tag{8.1}$$

The polynomial p is called the *interpolant*. Generally, if we have n data points, there is exactly one polynomial of degree at most $n-1$ going through all the data points in a certain order. The interpolation error is proportional to the distance between the data points to the power n. One advantage of this method is that the interpolant is a polynomial and thus infinitely differentiable in the range (x_0, x_n), with $x_0 < x_1 < ... < x_n$.

However, polynomial interpolation has some disadvantages. Calculating the interpolating polynomial can be computationally expensive for large n. Furthermore, polynomial interpolation may exhibit oscillatory artifacts, especially at the end points, the so-called Runge's phenomenon. This disadvantage can be avoided by using spline interpolation, which is presented later.

In the following, two particular cases of polynomial interpolation are considered, which solve the computation cost problem that appears for large n at the cost of increasing the interpolation error. The first, with $n = 0$, is the so-called *nearest-neighbor* or *piecewise constant* interpolation. The second, with $n = 1$, is the so-called *linear* interpolation.

Nearest-neighbor interpolation. The nearest-neighbor algorithm simply selects the value of the nearest sample, and does not consider the values of other neighboring samples at all, yielding a piecewise-constant interpolant. The algorithm is very simple to implement, and has been widely used in real-time 3D rendering to select color values for a textured surface. In this case, the differentiability property is lost.

Linear interpolation. Linear interpolation on a set of data points is defined as the concatenation of linear interpolants between each pair of data points. This results in a continuous curve, with a discontinuous derivative, thus of differentiability class C^0.

This particular case of polynomial interpolation is widely used in surface reconstruction, since it provides a sufficient level of accuracy for visualization and also for analysis, given a dense enough sampling of the surface, as it is shown in Chapter 10 in Part III. Furthermore, the resulting surface representation (a polygon mesh), can be processed by the standard rendering pipelines in graphics hardware. This type of interpolation is therefore the most suitable for interactive applications.

Spline interpolation

In the specific case of linear interpolation, a linear function for each of intervals $[x_k, x_{k+1}]$ is used as interpolant, whereas in the more general polynomial interpola-

tion a polynomial of high-degree n is used as the interpolant between a large number of data points or even the whole set of points. Spline interpolation uses low-degree polynomials in each of the intervals that would be used in linear interpolation, with the polynomial pieces chosen such that they smoothly fit in the resulting function.

Spline interpolation incurs a smaller error than linear interpolation and the resulting interpolant is smoother. It is also easier to evaluate than the high-degree polynomials used in the more general polynomial interpolation and does not suffer from Runge's phenomenon. However, the resulting interpolated surface, although showing a high level of accuracy for analysis, cannot be processed by the standard rendering pipelines in current graphics hardware with interactive visualization in mind.

Chosen methodology

A possible drawback in linear interpolation is the discontinuous derivative. However, given the fact that the input surface samples already contain information about the local orientation, this does not generate any practical limitation.

In the following, linear interpolation is taken as the most suitable option for its usage in analysis and visualization. Even though the interpolating error might be larger than that in spline-based interpolation, the possibility of using current graphics hardware for interactive visualization and the smaller computational cost associated to the extraction of the interpolant (a polygon mesh) makes us decide for this approach. Furthermore, a *spline* interpolation of a surface could also be obtained from a polygon mesh [Krishnamurthy and Levoy, 1996] at a post-processing stage if required.

About the ordering of the independent variable –*i.e.* the positions in 3D space of the surface samples–, different techniques exist in the literature, which produce linear interpolants which either pass exactly through the existing sample points or through close points defined by smoothing constraints on local sets of input samples. Some of these techniques are introduced in the next section, which are used to contextualize the necessity for the proposed method.

8.1.2 Related work

Many surface meshing techniques have been developed over the past years. The *marching cubes* (MC) algorithm was proposed long ago in [Lorensen and Cline, 1987]. This algorithm is a well-known method for extracting iso-surfaces from three-dimensional scalar fields. In practice, MC allows to extract a triangle mesh from a volumetric representation of a 3D scene.

Marching cubes-based methods

Many reconstruction techniques, like the ones presented in the first part of this thesis and others in the 3D reconstruction literature [Furukawa and Ponce, 2008], as well as range data from laser scans, do not provide a volumetric representation of 3D shapes, but rather a set of oriented surface samples. Hence, some approaches exist which, first, determine voxel occupancy by means of a certain type of regularization and, then, extract the surface by means of MC. Two state-of-the-art methods falling in this category are *RIMLS-MC* [Oztireli et al., 2009] and *Poisson-Rec* [Kazhdan et al., 2006].

RIMLS-MC. *Moving least squares* (MLS) is an attractive tool to design effective meshless surface representations. However, as long as approximations are performed in a least square sense, the resulting definitions remain sensitive to outliers, and smooth-out small or sharp features. In [Oztireli et al., 2009] these issues are considered and they present a novel point-based surface definition combining the simplicity of implicit MLS surfaces [Kolluri, 2005] with the strength of robust statistics, resulting in a *Robust Implicit Moving Least Squares* (RIMLS).

To reach this new definition, MLS surfaces are reviewed in terms of local kernel regression, opening the doors to the introduction of robustness. The authors claim that this representation can handle sparse sampling, it generates a continuous surface better preserving fine details, and can naturally handle any kind of sharp features with controllable sharpness. It also combines ease of implementation with performance competing with other non-robust approaches. Finally, a triangular mesh is extracted by using the Marching Cubes algorithm. RIMLS-MC has some desirable properties:

- It provides an accurate description of the surface, even in presence of noisy data.
- It shows better detail in edges, when compared to other marching cubes-based approaches.

However, it also presents some drawbacks:

- The method is relatively slow compared to other methods in the literature.
- The output surface, although topologically correct, can present holes, due to insufficient volumetric density.

Poisson-Rec. *Poisson Reconstruction* proposes a formulation that considers all the points at once, without resorting to heuristic spatial partitioning or blending,

and is therefore highly robust against data noise. As claimed by its authors, the Poisson approach presented in [Kazhdan et al., 2006] allows a hierarchy of locally supported basis functions, and therefore the solution reduces to a well conditioned sparse linear system. A spatially adaptive multi-scale algorithm is used, whose time and space complexities are proportional to the size of the reconstructed model.

In other words, surface reconstruction is expressed as a Poisson problem, which seeks the indicator function that best agrees with a set of possibly noisy, non-uniform observations. This approach can, for example, robustly recover fine detail from noisy real-world scans. At its last stage, an octree-based Marching cubes implementation extracts a triangular mesh from the volumetric representation. Therefore, PoissonRec has several desirable properties:

- It provides an accurate description of the surface, even in presence of noisy data.
- It allows choosing the level of detail in a trade-off with computational complexity.

However, it also presents some drawbacks:

- The precision around edges is not controlled.
- The output surface shows a limited amount of detail for similar computation times when compared to propagation methods.

Propagation methods

Propagation methods for surface reconstruction have in common the initial definition of one or more regions that are used as seeds for a propagation algorithm that attempts to cover the whole surface between the input 3D points. Two methods in the literature falling in this category are *Ball Pivoting* [Bernardini et al., 1999] and *ReOP* [Suau et al., 2010].

Ball Pivoting. The Ball Pivoting Algorithm (BPA), is an advancing-front algorithm to incrementally build an interpolating triangulation of a given point cloud. The method is conceptually simple. Starting with a seed triangle, it pivots a ball around each edge on the current mesh boundary until a new point is hit by the ball. The edge and point define a new triangle, which is added to the mesh, and the algorithm considers a new boundary edge for pivoting. BPA has several desirable properties:

- It is intuitive, since it triangulates a set of points by *rolling* a ball on the point cloud. The user chooses only a single parameter, its radius.
- It has a theoretical foundation, since it is related to alpha-shapes [Edelsbrunner and Mücke, 1994], and given sufficiently dense sampling, it is guaranteed to reconstruct a surface within a bounded distance from the original manifold.
- It keeps the positions of the original surface samples in the output mesh.

However, it also presents some drawbacks:

- It can produce surfaces with holes when sampling is not dense enough. In case of iterating with a larger radius for the pivoting ball, wrong topology might result from some local configurations.
- It is slow when compared to other methods, both based on a propagation scheme and on marching cubes.

ReOP. *Restricted and Oriented Propagation* is a method to reconstruct polygon meshes from oriented point sets based on a propagation through a voxelized space performed in order to find the closest neighbors of every data point. Propagation is done following a specific pattern which exploits reciprocity in neighbor finding. Every obtained pair of neighbors defines a new edge of the output mesh. ReOP has several desirable properties:

- It provides a quick reconstruction of the surface, which makes real-time applications conceivable.
- It keeps the positions of the original surface samples (or an average of 3D point cliques lying on the same voxel).

However, it also presents some drawbacks:

- The resulting mesh is composed by tetrahedrons rather than surface triangles, due to the 3D search strategy for neighbor points.
- The use of memory might be very high at high resolution, due to the necessity of regularly sampling space.

Goals

The goals of the method proposed in this chapter, compared to the features of those available in the literature, are:

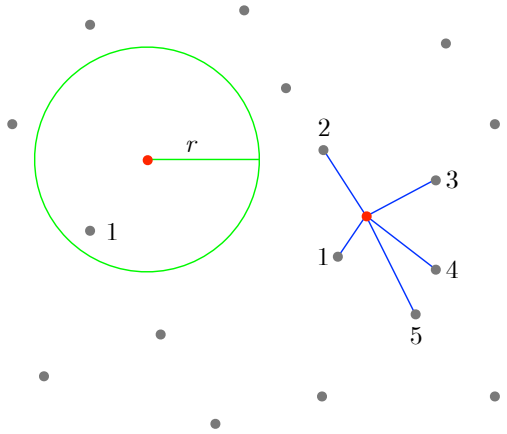


Figure 8.1: In green color, a range search with specified radius r around a given point –marked in red– returns a single result. In blue, a nearest-neighbors search with $k := 5$ returns a set of neighbor points around the the red point

- To preserve the position of the existing surface samples, like the propagation-based methods, in order to maximize the accuracy of the reconstructed surface when error-free data points are available. ReOP and BPA are therefore fulfilling this requirement.
- To obtain fast interpolation of surfaces with the goal of being usable in real-time applications. ReOP is fast, but does not guarantee topological correctness.
- To obtain a correct topology of the interpolated surface, making it usable for analysis applications. BPA, Poisson-Rec or RIMLS-MC can provide topologically correct surfaces, but are slow to be applied in real-time.

8.2 Methodology

The proposed algorithm follows the same principle as the ball pivoting algorithm, *i.e.* it is based in a surface propagation scheme, with the imposition of a set of rules to guarantee the topological correctness of the resulting surface.

First, the algorithm randomly selects a starting face and, then, it propagates on its contour, resulting in a growing region of triangular faces. The objective of the propagation phase is to find adjacent faces to, ideally, cover the whole surface with topologically correct triangles. As mentioned, a set of rules is checked whenever a face is to be added to the region, to ensure the resulting mesh is complete and manifold.

8.2.1 Spatial queries

Knowledge of local neighborhoods of each point and fast access to this information is a key factor of our proposal. A *kd-tree* decomposition of the 3D point cloud helps out in finding this information. As proposed in [Friedman et al., 1977], a kd-tree decomposition is an efficient structure –in terms of memory footprint and search speed– onto which *nearest-neighbors* and *range* queries are efficiently performed. The main concepts about this structure are presented in Appendix B, but here we focus on the two types of search –nearest-neighbors and range search– that are of interest for our proposed algorithm and are illustrated in Fig. 8.1. We recommend readers unfamiliar with this technique to look at Appendix B, in order to better understand the benefits of using such representation for efficient spatial queries.

In order to create the tree, instead of splitting the kd-tree in the axis of maximum spread as in [Friedman et al., 1977], we choose to split in that of maximum variance. This makes our the tree more robust against outliers in the sense that the efficiency of the search process is not affected by their presence. Other splitting rules might be applied, as suggested in [Maneewongvatana and Mount, 1999].

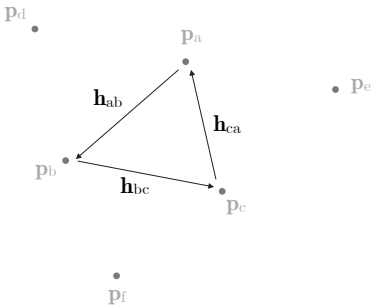
Nearest-neighbors search. In order to determine the composition of the neighborhood of a given surface point it is often necessary to obtain a list of the k nearest-neighbors to the given point. This is efficiently provided by a nearest-neighbors search in the kd-tree, as shown in Section B.2.1.

Range search. In some other cases, it is required to obtain all the points lying inside of a sphere of a determined radius centered at a given point. This search, which retrieves a list of points at a distance smaller or equal to r from the specified point, can also be efficiently implemented as a range search in a kd-tree, as shown in Section B.2.2.

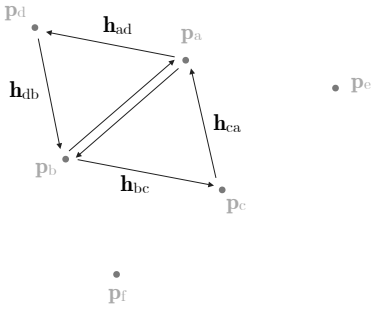
8.2.2 Half edges

In order to obtain a topologically correct polygon mesh, it is necessary to check whether a newly created face is compatible with the set of existing ones. Considering triangular faces, with three edges forming the contour of each face, it is necessary to ensure that the two new edges created after each propagation step are compatible with the already existing edges that may connect the involved vertices.

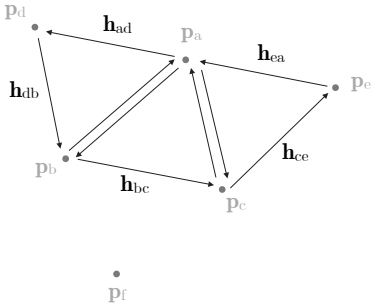
One useful representation of the polygon mesh is the so-called *half-edge* structure [Kettner, 1999]. In this structure, only edges are represented, which connect the vertices of the mesh in a certain orientation. In Fig. 8.2, the creation of a half-edge



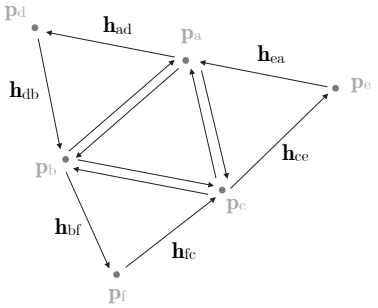
$$H^1 = \begin{pmatrix} 0 & \mathbf{h}_{ab} & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{h}_{bc} & 0 & 0 & 0 \\ \mathbf{h}_{ca} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$



$$H^2 = \begin{pmatrix} 0 & h_{ab} & 0 & \mathbf{h}_{ad} & 0 & 0 \\ h_{ba} & 0 & \mathbf{h}_{bc} & 0 & 0 & 0 \\ \mathbf{h}_{ca} & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{h}_{db} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$



$$H^3 = \begin{pmatrix} 0 & h_{ab} & h_{ac} & \mathbf{h}_{ad} & 0 & 0 \\ h_{ba} & 0 & \mathbf{h}_{bc} & 0 & 0 & 0 \\ h_{ca} & 0 & 0 & 0 & \mathbf{h}_{ce} & 0 \\ 0 & \mathbf{h}_{db} & 0 & 0 & 0 & 0 \\ \mathbf{h}_{ea} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$



$$H^4 = \begin{pmatrix} 0 & h_{ab} & h_{ac} & \mathbf{h}_{ad} & 0 & 0 \\ h_{ba} & 0 & h_{bc} & 0 & 0 & \mathbf{h}_{bf} \\ h_{ca} & h_{cb} & 0 & 0 & \mathbf{h}_{ce} & 0 \\ 0 & \mathbf{h}_{db} & 0 & 0 & 0 & 0 \\ \mathbf{h}_{ea} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{h}_{fc} & 0 & 0 & 0 \end{pmatrix}$$

Figure 8.2: Evolution of a set of half-edges after the addition of a new triangular face at every step. The current contour half edges at each step are named as $\mathbf{h}_{ij}$, where the subscript indicates the source and sink sample points that are connected by the half edge. The corresponding sparse matrix H^t after every step t shows the set of current half edges. The ones that continue the propagation are those left in bold in the matrices

representation and the corresponding contour at each propagation stage is shown.

The half-edge representation of meshes used by our method is stored as a sparse matrix H where all elements are empty, except for those that correspond to an active half edge, as shown in Fig. 8.2. These occupied elements are the only ones that are actually stored in memory.

The labeling of each non-zero element of the matrix is set to the identifier of the triangular face that generated the half-edge, which is used for correcting the topology of the mesh, as presented later in Section 8.3.2.

8.3 Algorithm

The algorithm presented in the following returns a polygon mesh that can be used as a linear interpolant of the surface between a set of input surface samples. In fact, the goal is to define the connections between close samples, or the ordering of the independent variable –*i.e.* the 3D position of the surface samples–.

The description of the resulting polygon mesh is such that it suffices at providing the required information for both visualization and analysis, without requiring additional information. As presented in Section 3.1.2, the polygon mesh will be represented as a list of vertices, each with its 3D position –and its RGB color, when available–, and a list of faces, each represented by three vertex identifiers.

The problem of finding the set of faces that correctly connect the input points is tackled in a region propagation fashion, as introduced above. First, a robust initial surface region –a triangle– will be defined and, then, the propagation on the current surface contour will start from the triangle's contour and iteratively end up in the covering of the whole surface.

8.3.1 Initial triangle

The choice of a starting face for propagation is of great importance, since it sets where the surface will start growing from. Thus, a set of conditions are required to be fulfilled in the form of a set of rules, in order to ensure a correct subsequent propagation.

A random unused starting point A and its k nearest-neighbors C_i are queried to the kd-tree, where C_1 and C_k are the closest and the farthest points in the returned set, respectively. The objective is to find a triangle between A, $B = C_j$ in the range $j = 1..k$ and a third valid C_i in the range $i = j..k$. In order to do so, points C_i must verify the following conditions (also depicted in Fig. 8.3):

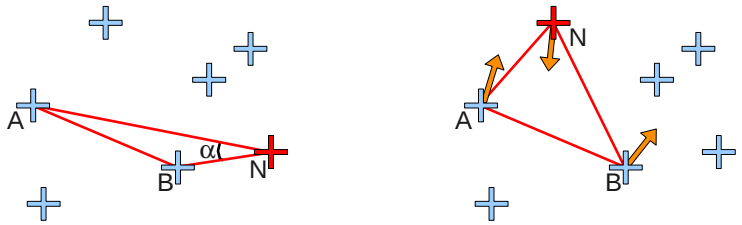


Figure 8.3: Rules for initial triangle creation. Left, situation where the *shape rule* (S) is not fulfilled. Right, situation where the *orientation rule* (O) is not fulfilled. Red color means *rejected*, green means *new* and grey means *already there*

- **(S) Shape rule**: Angle α_i at C_i must verify that $\alpha_{min} < \alpha_i < \alpha_{max}$.
- **(O) Orientation rule**: The normals of the three points, i.e. $\hat{\mathbf{n}}_A, \hat{\mathbf{n}}_B$ and $\hat{\mathbf{n}}_{C_i}$, must be coherently oriented.

If it is not possible to find a correct configuration from the randomly chosen point A and its neighbors, the method is restarted with a new randomly chosen point A. The first $C = C_i$ point which verifies the shape rule (S) and the orientation rule (O) with respect to two close points A and B is kept, obtaining an $F_1 = \widehat{ABC}$ starting triangle. The half-edge structure presented above is also updated with the three new half-edges just created, which are also added to a queue of contour edges that will be used throughout the propagation. In an abuse of notation, an *edge* in the queue actually corresponds to a *half-edge* in the former structure.

8.3.2 Propagation

Propagation starts at the initial triangle F_1, its three edges being the first contour edges to be processed. Contours to be propagated are extracted from a FIFO queue of contour edges. Then, it is checked whether they are active contours –*i.e.* the half-edge structure does not contain the opposite-oriented half-edge–. The idea is to expand contours with faces, while preserving topological properties at the same time. We define the following features related to a contour edge:

- F: Face the edge belongs to, with normal $\hat{\mathbf{n}}_f$.
- V_1, V_2 : Edge's start (source) and end (sink) vertices, respectively.
- V_3: Opposite vertex which completes the existing F.
- M_e: Edge's middle point, or $M_e = \frac{1}{2}(V_1 + V_2)$.
- $\hat{\mathbf{d}}_e$: Propagation direction from M_e, defined as $\hat{\mathbf{d}}_e = \hat{\mathbf{n}}_f \times (V_2 - V_1)$.

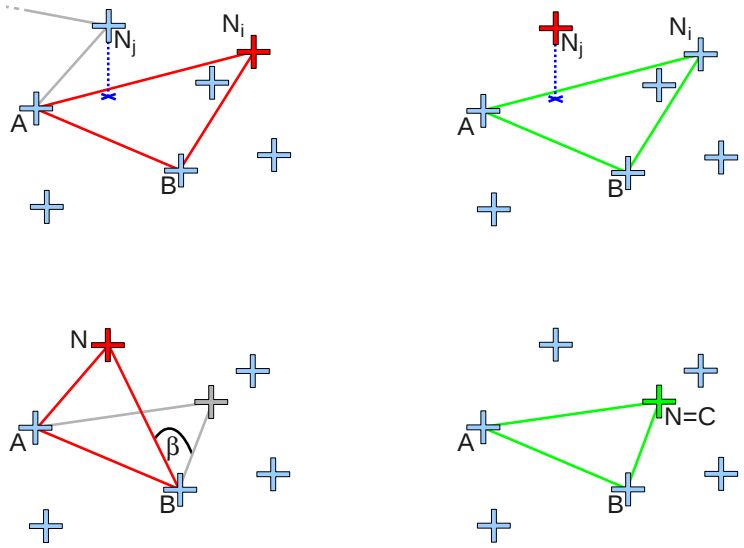


Figure 8.4: Rules for triangle propagation, added to those in Fig. 8.3. Top left, situation where the *used vertex inside rule* (U) is not fulfilled. Top right, situation where the *unused vertex inside rule* (UN) produces the rejection of such vertex. Bottom left, situation where the *spatial position rule* (P) is not fulfilled. Bottom right, a valid face is created, which fulfills the complete set of propagation rules. Red color means *rejected*, green means *new* and gray means *already there*

The k-nearest neighbors of M_e are queried to the kd-tree. Such vertices, called candidates or C_i, are processed in increasing distance order, beginning with the closest one C_1. In addition to rules for correct shape (S) and orientation (O), a candidate C_i (and the new candidate face $\tilde{F} = \widehat{V_1V_2C_i}$) must satisfy four more rules to be considered valid (see Fig. 8.4). The first two of these additional rules are the following:

- **(P) Spatial position** : The candidate vertex must be placed in a $\beta = 2\pi$ sr solid angle centered at the edge's middle point M_e and oriented as the propagation direction $\hat{\mathbf{d}}_e$.
- **(C) Contour or unused** : The candidate vertex must be either a non-used vertex or a contour one.

For the two last additional rules, we make use of the second type of search implemented for the kd-tree structure. Since it deals with a local neighborhood around each created face, a range search with a suitable radius is performed, in order to find all the existing points regardless of the sampling density. The idea behind these rules is to prevent the creation of two close layers of triangle faces coherently oriented. This might occur due to the existence of points distributed in two close layers when working with noisy surface samples.

- **(U) Used vertex inside** : Let bc_f be the baricenter of the new face $\tilde{F}$, and

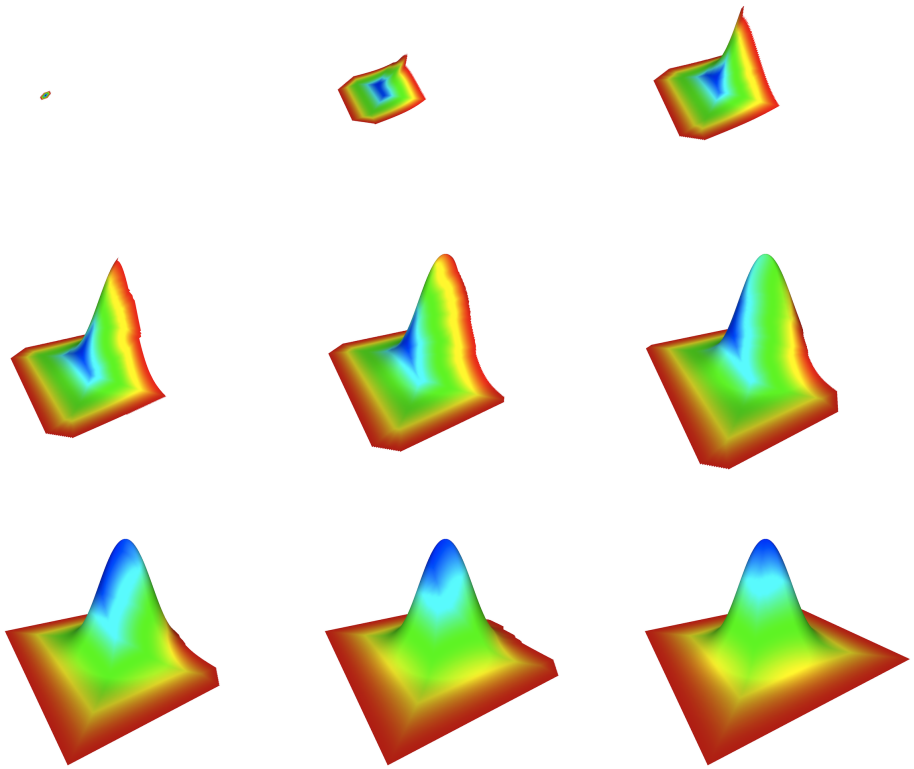


Figure 8.5: From left to right and top to bottom, propagation of a contour until completely covering the set of samples for a surface composed as samples of a bi-dimensional Gaussian function

$\mathcal{B}$ a ball centered at bc_f of radius $\rho = \max\{dist(p, bc_f)\}$ with $p = \{V_1, V_2, C_i\}$. If there exists an already used vertex $v_u \in \mathcal{B}$, which can be perpendicularly projected onto $\tilde{F}$, the candidate point C_i is rejected.

- **(UN) Unused vertex inside** : In the (U) context, if there exists an unused vertex $v_{un} \in \mathcal{B}$ which can be perpendicularly projected onto $\tilde{F}$, then v_{un} is rejected.

When a candidate point C_i, and its associated candidate face $\tilde{F}$, fulfill all of the preceding rules, this new triangle face is created. The queue of contour edges is updated with the newly added edges of $\tilde{F}$ and the half-edge matrix also reflects the introduced changes. In case that no candidate satisfies the imposed rules, the current edge is marked as *contour* edge and it is not further processed.

The propagation process is illustrated in Fig. 8.5, where the contour –colored in red– is iteratively propagated until covering all the samples of a surface corresponding to a bi-dimensional Gaussian surface.

Non-manifold edge detection

Despite of the set of rules for topological correctness described above, in some cases a wrong set of triangular faces might be created in some regions where different fronts of the surface propagation meet. This situation is illustrated in Fig. 8.6 (a). The gray region corresponds to a hole that is to be tessellated by propagating any of the contours around it. In Fig. 8.6 (c), two contour edges have propagated to new faces fulfilling all of the correctness rules presented above. However, this results in an incorrect configuration of overlapping surfaces.

From Fig. 8.6 (f) on, the mechanism for detection of a non-manifold edge and its posterior correction is illustrated. When a new candidate face $\tilde{F}$ is to be created, it is checked whether each of its contour half-edges can be correctly created –meaning that it does not exist in the half-edge structure–.

If one of the half-edges already exists, it is detected as a non-manifold edge. Consequently, the candidate face $\tilde{F}$ and the faces sharing the colliding half-edge –face identifiers are stored in the half-edge structure– are deleted as shown in Fig. 8.6 (g).

The half-edges that are set free by the removal of these faces adjacent to the non-manifold edge are added to the contour queue. Then, the normal propagation proceeds, which eventually produces a correct configuration by considering a reduced set of candidate points.

Iterative propagation

Propagation stops when no more contour edges are left to be processed in the queue. However, there may still be many unused points left. Iterating the initial triangle and propagation steps presents two advantages. First, it helps to achieve a complete meshing of the surface by filling zones which, following the rules for topological correctness, were not reachable from the previous starting triangles. Second, the sampled scene may contain many unconnected surfaces to be interpolated. Letting the system find new starting triangles will result in an individual meshing of each object.

Thus, when several disconnected surfaces compose the sampled scene, a new starting triangle is searched among the remaining unused points, which will launch a new propagation phase. Such cycle may be repeated iteratively while the amount of used points N_u is below a threshold. The propagation implementation used through this thesis has been stopped in all cases when $N_u > 90\%$.

In case that the propagation to a certain part of a surface has not taken place

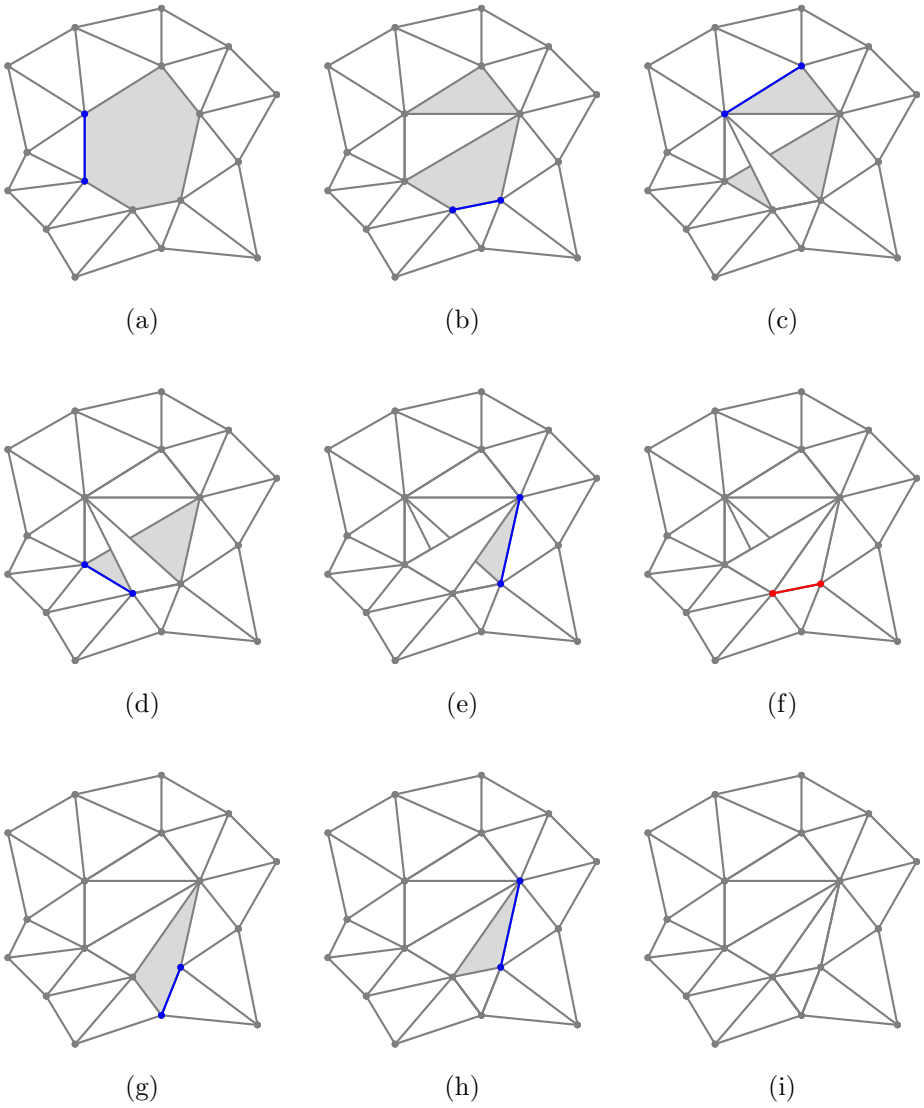


Figure 8.6: Incorrect surface interpolation and posterior correction. In every stage, the blue line signals the next contour edge to be processed in a queue. (a) An uncovered region (light gray) that is to be covered by the triangular mesh; (b) A half-edge propagates the surface following the set of rules for topological correctness; (c) another half-edge propagates the surface incorrectly, despite of following the set of rules for topological correctness; (d) and (e) two other contour half-edges propagate; (f) another contour half-edge propagates, in this case producing an overlap with an existing half-edge; (g) the three facets adjacent to the non-manifold edge are deleted; (h) and (i) the propagation proceeds without topological errors

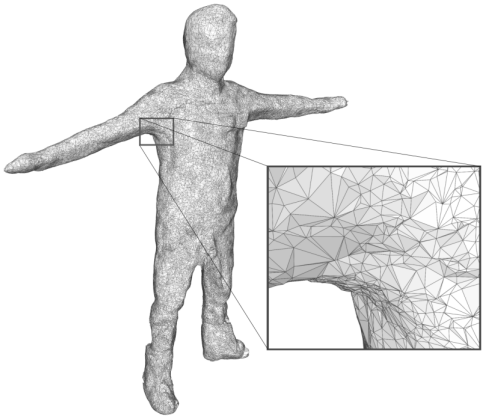


Figure 8.7: Topologically correct output mesh and detail of a challenging region around a saddle point

due to limitations imposed by the spatial position rule (P), the propagation would also proceed correctly, since every propagation phase takes into account the connectivities found during the previous propagation steps.

8.4 Conclusions

In this chapter, a novel meshing algorithm has been presented, which is useful to interpolate a continuous 2D surface in 3D space between samples corresponding to a discrete representation. It is driven by the main goal of defining an ordering for the input surface samples which is useful for interpolating the values of a continuous surface between these samples within a limited computation time.

The proposed algorithm, which has some points in common with the ball pivoting algorithm [Bernardini et al., 1999], being the main one its propagation-based principle, has accomplished its goals:

- It preserves the position of the existing surface samples, and is therefore capable of maximizing the accuracy of the reconstructed surface when error-free data points are available.
- It provides fast interpolations of surfaces with the use of an efficient set of data structures for spatial queries.
- It is capable to obtain a correct orientable surface, through the application of several rules for topological correctness and the possibility of automatically recovering from errors beyond the scope of these rules.

When compared to the ball pivoting algorithm, the proposed method does not require an iterative increment of the search radius in order to cope with scenes with

irregular distribution of surface samples. Instead, the use of *kd*-trees as a tool for spatial queries automatically isolates the meshing strategy from this problem. As illustrated in Fig. 8.7, the proposed method can reconstruct topologically correct surfaces even in cases where the input set of surface samples are unevenly distributed and represent a challenging geometric region such as a saddle point. More details about the performance of the meshing algorithm are presented in the next chapter.

As a difference with respect to volumetric algorithms like those based on the final extraction of the surface by marching cubes –*e.g.* [Kazhdan et al., 2006], a robust, state-of-the-art approach–, the chosen propagation scheme keeps unmodified the position of existing surface samples. This feature allows, for example, to easily interpolate the values contained in the surface samples to any position over the resulting continuous surface, using classical algorithms such as [Gouraud, 1971]. This can be done quickly thanks to the one-to-one correspondence between discrete surface samples and vertices in the mesh, which translates in a fulfillment of the main goal of the proposed method.